Kinetics of Oxygen Transport in Monomeric Sarcosine Oxidase

A Thesis

Submitted to the Faculty

of

Drexel University

by

Anthony Joseph Bucci

in partial fulfillment of the

requirements for the degree

of

Doctor of Philosophy in

March 2016

Dedications

My thesis is dedicated to my grandfather, Anthony Falco, who has always stood by me and set the example for the man I want to be.

Acknowledgments

First and foremost, I would first like to thank my advisor Dr. Cameron Abrams for his support, motivation, and patience. His dedication and enthusiasm have been a significant inspiration.

I would also like to thank my committee members Dr. Jason Baxter, Dr. Ken Lau, Dr. Raj Mutharasan, and Dr. Eric Vanden-Eijnden for their feedback and support.

I would also like to thank the Abrams' research group as well as our collaborators from the Courant Institute for Mathematics at New York University: Dr. Sergio Paz, Dr. Changwoon Jang, Dr. Francesca Moraca, Dr. Jungho Yang, Ryan Gordon, Jasmine Gardner, Arun Shrikant Sridhar, Dr. Eric Vanden-Eijnden, and especially Dr. Tang Qing-Yu for their advice and camaraderie.

Finally, I would like to thank my saint of a wife Michelle as well as my family, Mom, Tony, Dad, Grandpa, Arielle, Desiree, Karen, Russ, Christine, and Winnie for their love and unconditional support.

Financial support from the National Institutes of Health (grant No. R01GM100472) is acknowledged. This research was supported in part by the National Science Foundation through the TACC Supercomputing Center using XSEDE allocation (TG-MCB070073N).

Table of Contents

List of Tables

List of Figures

Abstract

Kinetics of Oxygen Transport in Monomeric Sarcosine Oxidase

Anthony Joseph Bucci

Advisor: Cameron Abrams, Ph.D.

Flavin-containing oxidases are a class of proteins which use oxygen to regenerate the oxidized state of the isoalloxazine ring in flavin adenine dinucleotide (FAD) or flavin mononucleotide (FMN) after it has been reduced by substrate oxidation. Though well characterized experimentally, many questions linger regarding how small molecules access the active site as well as where oxygen activation occurs in flavin-containing oxidases. A prototypical member of this family is monomeric sarcosine oxidase (MSOX), and it is perhaps the most well studied. Despite knowing which features are essential for catalysis as well as the location of the substrate binding site, it is unclear how oxygen accesses the site since the only clear entryway can be partially blocked by the larger substrate sarcosine. As such, two competing mechanisms have gained attention centering around the order by which ligands enter the binding site.

In this thesis, we detail the use of all atom molecular dynamics (MD) studies to identify how oxygen accesses the MSOX active site, as well as characterize the resulting kinetic network. We use the single sweep method to identify four potential routes for oxygen to travel from the surface of MSOX to the active site. Then, using Markovian milestoning with Voronoi tesselations (MMVT), we refine the pathways identified in single sweep and develop a Markov state model describing the kinetics of oxygen entry and exit. We calculate entry and exit mean first passage times (MFPT) for oxygen from this model, which are used to compute second order rate constants for entry and first order rate constants for exit. Our calculated rate constants and

mechanisms show that the presence of a substrate-mimicking inhibitor markedly influences the kinetics of O_2 entry and exit. The bound competitive inhibitor changes the protein structure sufficiently to shut down almost all major O_2 channels save one, which it opens, speeding entry but greatly slowing down O_2 exit, relative to the substrate-free enzyme. This means that our kinetic analysis predicts oxygen exhibits a longer residence time within MSOX when a substrate-like ligand is present. This supports the so-called "modified ping pong" mechanism, in agreement with previous experimental results, thus lending validity to our approach. Furthermore, our computed second-order entry rate constants are larger by about an order of magnitude than are experimentally determined O_2 consumption rate constants. Since O_2 consumption combines the processes of entry and electron transfer, we conclude that of these two, entry is not rate-limiting in the overall catalytic cycle, regardless of whether or not a substrate-like ligand is bound. Finally, because this work represents the first test of the MMVT approach for comparing kinetics of ligand entry into an enzyme in two distinct states, we not surprisingly uncovered many inefficiencies in the approach. We tested one idea for gaining efficiency based on the "finite-temperature" string method, in which transport channels can be determined and kinetically characterized on-the-fly, rather than sequentially. Our results indicate that more research in that area is needed.

1. Introduction

Aerobic processes in living organisms utilize proteins which can bind, react, and transport molecular oxygen. Myoglobin (Mb) for example, binds a single oxygen for use in muscular function. Hemoglobin (Hb) binds up to four oxygen molecules for transport within the pulmonary system. Other proteins, specifically flavin containing oxidases, use oxygen to regenerate the oxidized state of the isoalloxazine ring in flavin adenine dinucleotide (FAD) or flavin mononucleotide (FMN) after it has been reduced by substrate oxidation. Despite the range of functions for the systems described, the sequence of events remains the same. Oxygen must diffuse from the bulk solvent to the surface and enter, travel to a site where it can bind and possibly react, adsorb to the site, react, or if not, desorb, then travel back to the surface and ultimately exit into the bulk solvent. For the three cases previously mentioned, a wealth of experimental data exists, yet several key questions remain unanswered. These questions center around the routes small molecules like oxygen take to the active site, and how they are affected by binding of other ligands or substrates. Mb, the most studied of any protein, has only recently revealed its transport routes for CO [2]. This thesis will detail the use of all-atom molecular dynamics (MD) based techniques which identify routes oxygen utilizes in the flavin-containing oxidase monomeric sarcosine oxidase (MSOX). We will then use these routes as the basis for describing the resulting kinetic network and how it is affected by binding of a competitive inhibitor. The results indicate an accurate, robust, and efficient way to map small molecule transport pathways and characterize the associated entry and exit kinetics in ways which can be compared directly to experiment.

1.1 MSOX Overview

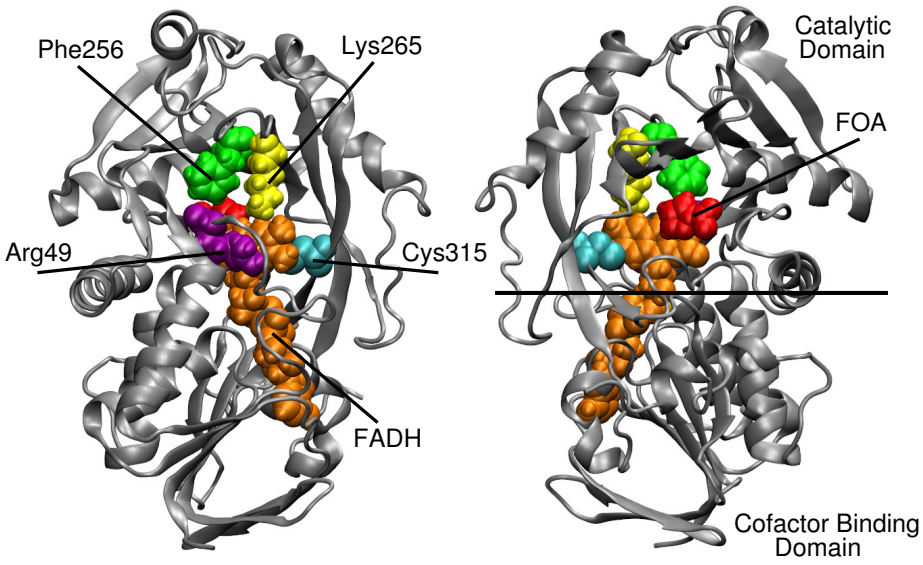

Figure 1.1: (Left) View of the *si*-face and (Right) *re*-face of the flavin isoalloxazine ring. The protein backbone is shown as a cartoon representation in silver. Important features and residues are shown in a space filling representation and colored individually.

MSOX belongs to a family of oxidases referred to as flavoprotein D-amino acid oxidases. These proteins reduce O_2 to H_2O_2 in order to regenerate the substrate-reactive oxidized state of their covalently bound flavin cofactors. Flavin oxidation by O_2 reduction is slow in free solution because of the requirement for an intermediate step in what would otherwise be a spin-forbidden two-electron transition [3]. Flavoprotein oxidases can speed this reaction up several orders of magnitude, although there does not appear to be a consensus mechanism [4]. Nevertheless, detailed structure-based studies suggest the importance of (a) the protein environment near the flavin for possible stabilization of intermediates, and (b) the protein structure in channeling O_2 from solution to the flavin site [5].

MSOX is probably the best-characterized flavoprotein oxidase in terms of structure and activity [1, 6–12]. It is a bacterial enzyme which catalyzes the oxidation of sarcosine (*N*-methylglycine) to produce formaldehyde, glycine, and hydrogen peroxide:

$$CH_3NHCH_2COOH + O_2 + H_2O \rightarrow HCHO + NH_2CH_2COOH + H_2O_2$$

MSOX is a 44 kDa two-domain protein consisting of 385 residues and contains a single covalently bound flavin adenine dinucleotide (8α-S-cysteinyl-FAD) whose flavin ring sits at the interface between the so-called catalytic and flavin domains. The flavin domain contains a cleft that admits and shelters the cofactor from solution.

Lys265 (Figure 1.1 yellow) is the catalytically active residue, but several other residues have been shown to be essential as well. Cys315 (Figure 1.1 cyan) covalently links the cofactor (Figure 1.1 orange) to MSOX. Severing of this link was shown to shut down catalytic activity [13], likely due to increased fluctuations of the isoalloxazine ring in the active site as well as the possibility for the cofactor to dissociate from MSOX. Arg49 (Figure 1.1 purple) lies across the *si*-face of the isoalloxazine ring. A basic residue at position 49 is required for covalent flavin attachment. However, mutation to Lys causes a 40-fold decrease in k_{cat} [10]. Thus, one may speculate that nature has optimized MSOX to have Arg at the 49 position. More recently, the need for a large nonpolar residue to stabilize the superoxide charge-transfer intermediate in flavoproteins has been explored [14]. In MSOX this large nonpolar residue is likely Phe256 (Figure 1.1 green). We observe in our work a large hydrophobic pocket on the *re*-face which exhibits Phe256 as the only large nonpolar residue near the active site. Lastly, there is the inhibitor FOA (Figure 1.1 red). Neither the native substrate sarcosine nor molecular oxygen readily co-crystallizes with MSOX, so we do not have definitive structure information on such states. Instead, researchers have used the

competitive inhibitor FOA to simulate binding of sarcosine in the active site [15] or chlorine to probe oxygen binding sites [12].

1.2 The Oxygen Activation Site

A major question regarding the O_2 reactivity of MSOX (and all flavoprotein oxidases) is the location of O_2 activation relative to the flavin ring and surrounding sidechains. The prevailing theory is that the two-electron reduction of O_2 passes through a superoxide anion intermediate on one of the two faces of the flavin isoalloxazine ring. This hypothesis has gained support because there are positively charged side chains near the flavin isoalloxazine ring [14]. X-ray crystal structures show the Lys265 ϵ-amino oriented over the *si*-face of the flavin ring with a bridging water (not shown) linking it to the flavin ring N(5) [6]. Lys265 is indeed crucial for O_2 reduction: mutations to Ala, Gln, or Met at position 265 result in an 8000-fold decrease in oxygen reactivity, and mutation to Arg results in a 250-fold decrease [9, 11]. It has therefore been proposed that sarcosine oxidation and oxygen reduction may occur on the *re*- and *si*-faces of the flavin ring, respectively [11, 12], which is consistent with the so-called modified ping-pong mechanism wherein O_2 can oxidize FADH- before the product imine is released [7].

However, one potential complication with this interpretation is that it is based on structures containing the oxidized flavin, while it is the reduced anionic form, with a protonated N(5) and a negative charge on N(1), with which O_2 must react. It is not clear that precise crystallographic arrangment of the Lys265 side chain and associated waters is retained with reduced flavin, which leaves unanswered the question as to whether or not this arrangement is a result of flavin oxidation rather than its cause. Thus, the question of where does oxygen activation occur has yet to be definitively answered.

1.3 Transport and Full Kinetic Network for O_2

A second major question regards how O_2 accesses the flavin cavity from the solvent. More specifically, what channels are transiently present due to protein fluctuations, and what is the kinetic network formed by their interactions? In the absence of bound substrate, there is clearly room for a single O_2 to reach the *re*-face of the flavin ring based on the crystal structure. However, the modified ping-pong mechanism suggests that O_2 may access the flavin while substrate is bound. Addition of a substrate partially blocks the channel observed in the static crystal structure, suggesting O_2 may access the cavity through separate channels. No such channels are readily apparent in the MSOX crystal structures. Since small dissolved gases access protein interiors through relatively small, fluctuating channels (e.g., the histidine gate in myoglobin [16, 17]), it is plausible that enzymes which utilize gas molecules as reactants may also exhibit such channels. Understanding the channels and their effects on the rate by which oxygen accesses the active site would provide important new structural insight into the biochemistry of flavoprotein oxidases and may even provide clues to the location of the O_2 activation site.

1.4 How Can We Answer These Questions?

We can begin to answer these questions by querying the statistical likelihood of observing an O_2 molecule at any position inside the protein. In similar settings, this has been approached using analysis of static crystal structures and all-atom molecular dynamics (MD) simulations, as well as enhanced sampling variants thereof. In all such efforts, an accurate picture of the accessible volume in the protein interior is limited by inadequate sampling of protein conformations. This is a severe problem in sampling static structures [18, 19] and even in brute-force molecular dynamics (MD) [20, 21], especially if one needs enough statistics to map out the entire protein

interior. Therefore, enhanced-sampling methods remain an imporant tool for this kind of mapping. For instance, Elber applied the time-dependent Hartree approximation to study CO diffusion in myoglobin [22]. Ceccarelli et al. used the metadynamics method [23] to study the same system [24]. Saam et al. applied implicit ligand sampling (ILS) to study dynamic oxygen access channels in 12/15-lipoxygenase [25]. Recently, a composite method combining temperature-accelerated MD [26], the single-sweep method of free-energy reconstruction [27], and the string method in collective variables [28] was used to fully map diffusion pathways and adsorption sites for CO [29] and water [30] in the interior of myoglobin, including thermodynamic characterization of the pathways. We will employ this composite method to overcome the obstacle of limited sampling previously encountered in similar work.

However, none of these approaches provide any reliable kinetic rate data to compare against experiment. Previous molecular simulation work in this context has focused primarily on identifying possible routes O_2 takes between the bulk solvent and the active site in similar enzymes. For example, using MD and Implicit Ligand Sampling (ILS), Saam showed that O_2 diffusion through 12/15-lipoxygenase is coupled dynamically to side-chain reorientations in several channels connecting solvent to the active site [25]. Baron used enhanced statistics MD to show spontaneous diffusion of O_2 to the active site in the flavoenzymes *p*-hydroxyphenlyacetate hyroxylase and alditol oxidase [31]. More recently, Shadrina used Temperature Controlled Locally Enhanced Sampling (TLES) to show O_2 escape routes from the heme to the solvent based on the state of HisE7 in hemoglobin [32]. Additionally, in the cofactor independent oxygenase DpgC, Di Russo used MD enhanced with multiple GPUs to determine spontaneous O_2 diffusion routes to the active site [33]. Although these works support the idea that small gas molecules can access buried sites by multiple pathways, it remains challenging to determine which pathways contribute the most

to the rate at which O_2 accesses an active site, since none provide a direct way to calculate such rates. This makes these and similar methods somewhat limited when trying to understand fundamental kinetics.

To determine rates which can be compared directly to experiment, Yu et al. recently adapted the MD-based method of Markovian Milestoning [34] to compute rates of diffusion, entry, and exit of small molecules in proteins, and applied it to the study of CO entry and exit from myoglobin [2]. We will apply Markovian milestoning to the MSOX/O_2 system with the aim of addressing (a) whether or not the rate of flavin oxidation is limited by the diffusion of O_2 from the bulk solvent to the active site, and (b) whether O_2 reduction in that site occurs before or after release of the oxidized product. After calculating entry and exit rate constants, we can compare full O_2 transport kinetics in apo and inhibitor-bound structures of MSOX to the experimental results. In this way we can also assess whether our results support the ping-pong or the experimentally consistent modified ping-pong mechanism [1].

1.5 Aims of Thesis

1.5.1 Aim I: Determination of potential oxygen transport pathways and identification of the oxygen activation site

The composite approach of Maragliano et al. is applied here on MSOX to map O_2 sites and pathways to the substrate activation site. We first map the free energy of oxygen as a function of position in the interior of MSOX. Then, after identifying regions of low free energy, we determine minimum free energy pathways (MFEPS) connecting the active site to the solvent. Based on the location of regions of low free energy and the MFEPs determined from the composite method, we also attempt to find the oxygen activation site.

1.5.2 Aim II: Refinement of the kinetic network with Markovian milestoning and comparison to experimentally determined rate constants.

Using Markovian milestoning in voronoi tesselations (MMVT) we examine the MFEPs identified using the composite approach of Maragliano et al. MMVT will calculate mean first passage times (MFPTs) for the interconnected kinetic network. This will allow us to identify major and minor pathways, as well as how the network changes with binding of an inhibitor. The MFPTs will permit calculation of second order entry and first order exit rate constants for oxygen. We then compare our rate constants which are consistent with modified ping-pong to similar values obtained experimentally. The calculated second-order rate constants are faster than those determined experimentally. However, we only model entry and exit, while experimental results encompass entry, oxidation, and exit (in the event of unsuccessful oxidation). Our results thus predict that the process of O_2 entry does not limit its reduction rate.

2. Computational Methods

2.1 Overview

The purpose of this work is to calculate entry and exit rates for O_2 in MSOX. We will accomplish this task through a hierarchy of techniques based upon molecular dynamics (MD). We begin by building apo and bound systems for MSOX, and subjecting them to long MD. Here, we observe protein equilibration and a general lack of interior sampling by O_2. We then turn to the enhanced sampling technique Temperature Accelerated Molecular Dynamics (TAMD) to completely sample the interior of MSOX [26]. With a well sampled interior, we apply single sweep [27] to map the free energy of O_2 as a function of position within MSOX.

After mapping the interior, a search for regions of low free energy is performed via multiple walker steepest descent on the reconstructed free energy surface. Regions of low free energy indicate O_2 has a high probability of localizing there. In this way we identify the active site and tentative portals for entry and exit. At this point we can now determine minimum free energy pathways (MFEPs) which are assumed to carry the majority of the flux between any individual portal and the active site. MFEPs are identified via Zero Temperature String Method (ZTSM) [35].

At this point we have identified an initial geometrical network as determined by ZTSM and the reconstructed free energy surface. The next step is to calculate entry and exit mean first passage times (MFPTs) from the active site to any surface portal, and ultimately the nearby solvent. We apply Voronoi-cell milestoning [34] to determine the MFPTs and arrive at a final kinetic network. Lastly, with MFPTs available, we approximate second-order rate constants for entry and first-order rate constants for exit to compare against experiment.

2.2 Molecular Dynamics

Molecular dynamics (MD) is a simulation technique based on classical mechanics where, given initial atomic positions and their velocities, velocities and positions at the next time-step can be calculated from numerical integration of Newton's equations of motion. This is achieved by allowing each atom within the system to interact via a force field. The force field is the negative gradient of an interaction potential accounting for the interactions occurring within the system. These include bonds, angles, dihedrals, van der Waals, and electrostatics. After forces have been computed for a given configuration of positions, numerical integration of Newton's equation of motion generates new positions and velocities. The new positions and velocites are now advanced one time-step, typically $1-2$ fs. Depending on available computational resources, this process can repeat billions of times, resulting in a trajectory on the scale of up to μs with current technology.

2.3 MD Setup and Simulations

The primary technique in this work is all-atom molecular dynamics (MD) underlying a particular series of free-energy calculations or transition path theory (TPT) simulations. All-atom systems of inhibitor-bound and apo MSOX were initially prepared. Heavy-atom coordinates for MSOX were taken from the 2gf3 PDB entry [15]. In addition to the protein, this structure contains several waters and the inhibitor 2-furoic acid (FOA), the latter of which competitively binds at the active site. The FOA-bound MD system was generated by adding hydrogens to the 2gf3 coordinates where needed, solvating in TIP3P water [36], neutralizing with Na^+ ions, to generate a box of 33,226 atoms with dimensions $83.8\times64.1\times67.6$ Å. This initial system was subject to 1000 steps of conjugate gradient minimization followed by 130 ns of NPT MD equilibration.

The apo system was created by deletion of the FOA from this minimized system and subjecting the resulting system to 130 ns of NPT equilibration. A molecule of O_2 was added by mutating one of the waters coordinating Lys265. This system consisted of 33,221 atoms in a box 83.7×64.0×67.5 Å. We did not observe spontaneous opening of the so-called active site loop (residues 55-60) in the apo equilibration, and all calculations reported here are with this loop in the closed position. It may exist in either open or closed configurations for apo MSOX but only closed for ligand-bound MSOX [11]. However, spontaneous opening of the FAD cleft was observed in the apo equilibration, distinguished by the C_α-C_α distance between Asn41 and Arg282. The distribution of C_α-C_α distances for this residue pair is centered at 10 Å when the cleft is open and 8 Å when closed. We consider both apo states in single sweep and ZTSM analysis.

Molecular dynamics simulations employed periodic boundary conditions, a non-bonded cutoff of 10 Å, a particle-mesh Ewald spacing of 2 Å, rigid bonds, and a timestep of 2 fs. The temperature was held at 310 K using a standard Langevin thermostat and 1 bar using a Langevin-Nosé-Hoover barostat [37]. Finally, because the subsequent free-energy calculations are performed in a protein-fixed coordinate frame, weak positional/rotational restraints were applied. These consisisted of cartesian harmonic restraints on the C_α's of residues 25, 100, and 370 with a common spring constant of 1 kcal/mol-Å^2. All simulations were conducted with NAMD v. 2.9 [38] using the CHARMM force field [39, 40].

Protonation of the flavin was made consistent with its reduced form, $FADH^-$. CHARMM-style parameters for the adenine and sugar portions of $FADH^-$ were adapted from existing parameters for NADH. The flavin ring was parameterized using the AMBER antechamber procedure with parameters, including charges determined independently using geometry optimization at the B3LYP 6-311G* level using Gaus-

sian, translated into CHARMM-style units. FOA was similarly parameterized. The parameter sets for $FADH^-$ and FOA were not further optimized. Charmm-style topology and parameter files for $FADH^-$ and FOA are available in Appendix E.

2.4 Interior Sampling and Temperature Accelerated Molecular Dynamics

We begin by completely sampling the MSOX interior structure using Temperature Accelerated Molecular Dynamics (TAMD). Generally, TAMD accelerates the sampling of collective variables (CV's) $\boldsymbol{\theta}(\boldsymbol{x})$ in an MD simulation by tethering them to fictitiously hot auxiliary variables $\boldsymbol{z}$ with high friction $\overline{\gamma}$, such that the forces these variables experience approximate negative gradients on the free-energy surface of the CV's [26]:

$$\overline{\gamma}\dot{z_j} = \kappa[\theta_j(x) - z_j] + \sqrt{2\overline{\beta}^{-1}\overline{\gamma}}\eta_j^z(t). \tag{2.1}$$

Here, κ is a spring-constant-like parameter, $\overline{\gamma}$ the artificial friction coefficient, $\overline{\beta}$ the inverse of the artificial temperature ($\beta \equiv 1/k_BT$, where k_B is Boltzmann's constant), and $\eta_j^z(t)$ is white noise with unit variance.

In the approach used by Maragliano et al. for CO in myoglobin, the CV's accelerated were the Cartesian coordinates of the CO center of mass [29]. Analogously, here the Cartesian coordinates of the O_2 center of mass are accelerated. Each TAMD simulation, or "sweep", used the same conditions as did the MD equilibrations, with the additional TAMD parameters of $\kappa = 200$ kcal/mol-Å and $\bar{\gamma} = 5$ ps^{-1}.

The goal of TAMD is to sample as much of the O_2-accessible volume in the protein as possible. Since three systems were studied, the following naming convention was adopted. Inhibitor bound MSOX is referred to as 'bound MSOX' while apo MSOX with the Asn41-Arg282 bridge open or closed 'apo-OB MSOX' and 'apo-CB MSOX' respectively. Initial sweeps for bound and apo-CB MSOX were run at ficitious temperatures $2 \leq \bar{\beta}^{-1} \leq 7$ kcal/mol. Three sweeps were run for each discrete

fictitious temperature at both the *re*-face near Lys265 and below the *si*-face of the flavin ring. Productive sampling, defined as sampling greater than that observed in MD and remaining inside MSOX for the majority of the sweep, was observed between 3 and 5 kcal/mol. Sampling comparable to MD was obtained below 3 kcal/mol and insufficient interior sampling achieved above 5 kcal/mol because O_2 exited rapidly, exploring little of the interior.

To achieve consistent protein interior volume sampling across the three systems, 1 ns sweeps were initialized from four seperate sites within MSOX, two in each domain. Initial production sweeps were run from two sites in the catalytic domain, the *re*-face near Lys265 and below the *si*-face of the flavin ring. Sites were required in the FAD binding domain because regions were accessible to only apo or bound sweeps. Therefore, additional production sweeps were launched from near Leu212 and C(5)′ on FADH but not in the cleft. For any site, a water was mutated into oxygen from the equilibrated starting structure. A total of 36 sweeps were required for apo-CB MSOX. Fifty-one sweeps were required for apo-OB, and 64 for bound MSOX. A total of 151 ns of production sweeps were required to achieve consistent interior volume sampling among all three systems.

2.5 Single Sweep Free Energy Reconstruction

The main idea of single-sweep is to use mean forces computed on a small number of important locations as the basis for reconstruction of a complete analytical free energy as a function of O_2 position in the protein. The output of the TAMD production sweeps is a dense set of O_2 positions both inside and outside the protein. Since we need only a small number of these points, composite trajectories were culled by beginning with an interior location and including any location not closer than 2.5 Å to any already-included location. Each harvested location is referred to as a “center”

in single sweep, and the kth center is indexed as $\boldsymbol{z}_k$. A total of 262 centers were harvested for apo-CB MSOX, 355 for apo-OB MSOX and 416 for bound MSOX.

At this point the small number of important locations needed to reconstruct the free energy have been chosen. We now must estimate the mean-force acting on each of the points through restrained MD simulations. Mean-force calculations for each center proceeded as follows. The MD system for any center is run under the TAMD protocol with the protein-restraints active and the fictitious friction effectively infinite, so the auxiliary variables do not evolve. The mean force vector $\boldsymbol{f}$ at center position $\boldsymbol{z}_k$ is computed as the following time-average on the atomic trajectory $\boldsymbol{x}(t)$ as:

$$\boldsymbol{f}(\boldsymbol{z}_k) \equiv \boldsymbol{f}_k = \frac{1}{T}\sum_{j=0}^{T} \kappa\left[\boldsymbol{\theta}(\boldsymbol{x}(t_j)) - \boldsymbol{z}_k\right] \tag{2.2}$$

where T is the number of time increments in the trajectory and $\boldsymbol{\theta}$ is the collective variable (instantaneous O_2 center of mass). Saturation of mean forces was observed to occur in less than 5 ns of MD integration using 1 fs time-steps.

The reconstructed free energy $\tilde{A}(\boldsymbol{z})$ is represented analytically as a radial-basis function expansion:

$$\tilde{A}(\boldsymbol{z}) = \sum_{k=1}^{K} a_k \varphi_\sigma(|\boldsymbol{z} - \boldsymbol{z}_k|) + C \tag{2.3}$$

where φ_σ is a Gaussian with width σ, a_k is the kth coefficient in the basis-function expansion, and C is an irrelevant constant that adjusts the overall height of $\tilde{A}(\boldsymbol{z})$. Via standard linear-algebra methods, the fitting parameters a_k and σ are determined by minimizing an error function given by:

$$E(a,\sigma) = \sum_{k=1}^{K} |\sum_{k'=1}^{K} a_{k'} \nabla_z \varphi_\sigma(|\boldsymbol{z}_k - \boldsymbol{z}_{k'}|) + \boldsymbol{f}_k|^2 \tag{2.4}$$

The optimal basis function width (σ) was 2.5 Å with a relative residual error of

approximately 0.66 kcal/mol/Å for all three configurations.

Once the free energy of oxygen as a function of position has been mapped, regions of low free energy (high probability for oxygen to localize) must be identified. These areas are referred to as local minima, and are located on the FES using multiple-walker steepest-descent minimization. For each center previously identified, a walker is allowed to "fall" into the nearest free energy minimum, thus identifying local minima. From the final walker locations, sets of interesting local minima are identified. To find pathways of minimum free energy between any two of these minima, the zero-temperature string method is used [35]. Briefly, for two minima A and B, a line segment connecting them is discretized into N sites, each of which is a walker that is allowed to move according to the local gradient in $\tilde{A}$ subject to a reparameterization step that keeps the site-site separation distance along the string uniform. The string of sites thus "falls" into a minimum free-energy path (MFEP) connecting the two minima. String convergence for this investigation was achieved when the change in free energy between successive calculations was approximately 10^{-5} kcal/mol. We typically used $N = 50$ discretization points on each string.

2.6 Milestoning

Milestoning generally refers to a method in which the rate of some rare event is computed through many independent MD simulations initiated at discrete locations along a reaction coordinate. These simulations evolve freely until transitioning to another location. The transitions provide input for the statistical analysis and subsequent rate calculations [41, 42].

Here, we apply Markovian Milestoning in Voronoi tesselations (MMVT), an application of transition-path theory (TPT) [34] to milestoning which increases accuracy and efficiency. MMVT has been presented in detail in the past, and the variant

we employ was applied in detail to our CO/Mb study [2]. We therefore present a summary of the method here.

The major steps in the method are as follows.

1. Compute $F(\boldsymbol{z})$, the free energy associated with the cartesian position of the center of mass of an O_2 molecule $\boldsymbol{z}$ in a coordinate system defined by the protein.

2. Analyze the function F to find local minima and minimum free energy pathways (MFEP's) interconnecting them and solvent portals.

3. Setting each string image defining a MFEP to be the center of a Voronoi cell, run MD confined within each cell.

4. Analyze MD results to determine the mean first passage times (MFPT's) for an O_2 to transit each pathway in either direction.

5. Incorporate a bulk diffusion model to compute second-order entry rate constants based on solvent-to-site MFPT's and bulk O_2 concentration.

We reported previously on steps 1 and 2 [43]. There we used the method of single-sweep reconstruction [27] to compute F for three different states of the MSOX protein which include two *apo* and one *bound* configuration. We will now apply Markovian milestoning beginning at step 3 to the three configurations previously identified.

2.6.1 Markovian Milestoning

The key conceptual requirement of milestoning is to represent a hypothetical infinitely long MD trajectory (which would contain all rare transitions that for practical reasons we can never observe) as a sequence of state transitions and lag times between transitions. The first key question then is, what are these "states"? We choose to use

the faces of a Voronoi tesselation of the chosen collective variable (CV) space. We generate a Voronoi tesselation by choosing discrete centers along all MFEP's, with the general rule that the first selection of centers produces a tesselation with a spacing between faces of about 0.67 Å. Let $\boldsymbol{x} \in \Omega \subset \mathbb{R}^d$ represent the coordinates of the system and $\boldsymbol{\theta}(\boldsymbol{x}) = (\theta_1(\boldsymbol{x}), \ldots, \theta_M(\boldsymbol{x}))$ are our collective variables. We define a set of points in CV space $\boldsymbol{z}_i \in \mathbb{R}^M$ with $i = 1, 2, 3..., \Lambda$, which partition the configuration space into Λ Voronoi cells. The Voronoi cell B_i at z_i is defined as all configurations whose mappings into CV space are closer to $\boldsymbol{z}_i$ than to any other center:

$$B_i = \{\boldsymbol{x} \in \Omega : \|\boldsymbol{\theta}(\boldsymbol{x}) - \boldsymbol{z}_i\| < \|\boldsymbol{\theta}(\boldsymbol{x}) - \boldsymbol{z}_j\| \text{ for all } j \neq i\}, \tag{2.5}$$

where $\|.\|$ is Euclidean distance. The face between any two adjacent Voronoi cells B_i and B_j is denoted S_{ij} and is termed a "milestone state". A transition generally can occur from milestone S_{ij} to S_{ik}, which interfaces cell B_i to cells B_j and B_k, respectively. Imagine now we invoke the existence of an overall transition matrix $\mathbf{Q}$ spanned by all milestone states, which reports the average rate of transition from any one state to any other. One could populate this matrix by extracting from the infinitely long MD simulation the following quantities: (i) $N_{ij,ik}$, which records the number of times that the trajectory collides with face S_{ik} after having last hit face S_{ij}, and (ii) R_{ij}, which records the total time the system spends in state ij, that is the total time in which S_{ij} was the last most recent milestone encountered. Via a maximum-likelihood estimate, the nondiagonal elements of $\mathbf{Q}$ are given by

$$q_{ij,ik} = \frac{N_{ij,ik}}{R_{ij}} \text{ if } R_{ij} \neq 0 \tag{2.6}$$

where $q_{ij,ik}$ is the rate from milestone state S_{ij} to S_{ik}. Milestones that are never visited (i.e., such that $R_{ij} = 0$) can be removed from the Markov chain. This reduces

the original configurational space dynamics to a jump process with rate matrix $\mathbf{Q}$. $\mathbf{Q}$ can be thought of as the basic objective of the milestoning procedure; once we have it, we can analyze it to describe the kinetics of our system using any convenient or meaningful renormalization of the base set of milestone states.

Now, we do not have an infinitely long MD trajectory, so we instead run short independent MD simulations confined to each cell. A simulation is assigned to cell B_i and constrained using reflective boundary conditions that are applied to *all* atoms in the system at the instant of a Voronoi violation:

$$x_i(t+\Delta t) = \begin{cases} x_i(t+\Delta t) & \text{if } x_i(t+\Delta t) \in B_i \\ x_i(t) & \text{otherwise,} \end{cases} \tag{2.7}$$

and

$$v_i(t+\Delta t) = \begin{cases} v_i(t+\Delta t) & \text{if } x_i(t+\Delta t) \in B_i \\ -v_i(t) & \text{otherwise.} \end{cases} \tag{2.8}$$

From TPT, we know the Voronoi faces are locally orthogonal to the MFEP's that connect local minima in F and are good approximations to *isocommittor surfaces*. The advantage to using Voronoi cell based milestoning is that we do not need to generate initial data on the milestones. This task is difficult as the data must be generated from a non-equilibrium distribution of hitting points, and then reinitialized after each iteration [41, 42]. Since the probability that a trajectory launched from any point on an isocommittor surface will reach the product state before returning to the reactant state is invariant, the initial location of O_2 in a particular cell is arbitrary. Avoiding the use of specific milestones is founded not only in the theory of milestoning, [44] but in further work on trajectory parallelization, [45] and non-equilibrium umbrella sampling (NEUS) [46]. These methods are exact in all cases.

To estimate $N_{ij,ik}$ and R_{ij}, the MD simulation in each cell B_i with total simulation time T_i is subject to the following analysis:

1. For each adjacent cell B_j, we record the number of attempted transitions across milestone state S_{ij} from within cell B_i, $N_{i\to j}$ and calculate the quantity $k_{i\to j} = N_{i\to j}/T_i$.

2. For all pairs of milestone states (S_{ij}, S_{ik}), we record the number of transitions from S_{ij} to S_{ik} while necessarily remaining within B_i, $N^i_{ij,ik}$, and calculate the quantity $n^i_{ij,ik} = N^i_{ij,ik}/T_i$.

3. For all S_{ij}, we record the time the confined trajectory accumulates after having hit S_{ij} before next hitting any other milestone state, R^i_{ij} and calculate the quantity $r^i_{ij} = R^i_{ij}/T_i$.

T_i is the total simulation time in cell B_i. $k_{i\to j}$, $n^i_{ij,ik}$, and r^i_{ij} should saturate to constant values as $T_i \to \infty$; in practice, the simulation in any cell B_i is run long enough to observe these saturations (normally between 5 and 50 ns for this system). $k_{i\to j}$ is the rate estimate for the system to escape from cell B_i to B_j. We require that the equilibrium probability π_i for the system to locate in cell B_i satisfies a balance equation:

$$\sum_{\substack{j=1\\ j\neq i}}^{\Lambda} \pi_j k_{j\to i} = \sum_{\substack{j=1\\ j\neq i}}^{\Lambda} \pi_i k_{i\to j} \ , \ \sum_{i=1}^{\Lambda} \pi_i = 1 \tag{2.9}$$

The solution gives π_i, and consequently the free energy for locating the system in cell B_i, $-k_B T \ln(\pi_i)$. The cell-simulation specific quantities $n^i_{ij,ik}$ and r^i_{ij} can then be used to evaluate elements of $\mathbf{Q}$:

$$q_{ij,ik} = \frac{\pi_i n^i_{ij,ik}}{\pi_i r^i_{ij} + \pi_j r^j_{ij}} \tag{2.10}$$

2.6.2 Ligand Entry

In the CO/Mb work, our group detailed a new method for handshaking bulk transport of the dissolved gas molecule with the milestoning description [2]. The basic idea of this method is that we graft a continuum-level description of the diffusion-limited flux into the milestoning framework with predefined "solvent milestones". First we identify, for each portal, the outermost cell along the MFEP that "leaks" to solvent; that is, a cell in which O_2 accesses the solvent without hitting a milestone. That cell is disregarded and the space defined by a sphere centered at the center of the next cell inward from that cell is defined and tesselated into a set of "portal cells". The size of the sphere is chosen such that it represents a boundary between bulk solvent and solvent within interaction distance of the protein atoms that define the portal. Milestoning MD is run in these and their data is included in the network of cells. The outermost set of these cells have milestones that interface the bulk solvent, which we label specially as "solvent milestones", S_{sj}.

The key feature of the method is an estimate of the total flux on all solvent milestones:

$$J = \frac{AD[O_2]}{R} \tag{2.11}$$

where $A = \sum_j A_{sj}$ is the total area of the solvent milestones, and A_{sj} is the area of solvent milestone S_{sj}, D is the bulk self-diffusion constant for O_2 in water at $37°C$, and $[O_2]$ is the bulk concentration of O_2. The flux $N_{s\to j}$ on each solvent milestone S_{sj} is proportional to the ratio between the area of this milestone and the total area of the solvent milestones,

$$N_{s\to j} = \frac{JA_{sj}}{A} = \frac{D[\mathrm{O_2}]A_{sj}}{R} \tag{2.12}$$

where A_{sj} is the area of solvent milestone S_{sj}. We can then use these fluxes to get all

cell probabilities, including that of the fictitious "cell" representing bulk solvent, π_s. The rate from a solvent milestone state S_{sj} to any inner, or non-solvent, milestone state, say, S_{jk} can be calculated from:

$$q_{sj,jk} = \frac{\pi_j n^j_{sj,jk}}{\pi_s r^s_{sj} + \pi_j r^j_{sj}}. \tag{2.13}$$

All quantities in this expression are known from the milestoning MD simulations except for the fraction of time it is assigned to milestone S_{sj}, r^s_{sj}. Since O_2 is assumed to freely diffuse in the solvent, we can further assume

$$r^s_{sj} = \frac{A_{sj}}{\sum_{sk} A_{sk}}. \tag{2.14}$$

For pairs of solvent-milestone states, we use a detailed balance to calculate π_i:

$$\pi_{sj} q_{sj,sl} = \pi_{sl} q_{sl,sj} \tag{2.15}$$

where $\pi_{sl} \propto (\pi_s r^s_{sl} + \pi_l r^l_{sl})$ and $\pi_{sj} \propto (\pi_s r^s_{sj} + \pi_j r^j_{sj})$. Thus, we should have

$$q_{sj,sl} = C(\pi_s r^s_{sl} + \pi_l r^l_{sl}) \ , \ q_{sl,sj} = C(\pi_i r^s_{sj} + \pi_j r^j_{sj}) \ , \ etc \tag{2.16}$$

for some constant C with dimension of a flux: here we will simply assume $C = J$.

2.6.3 TPT Analysis of a Markov Jump Process

The final piece of the method is the extraction of meaningful rate constants and MFPT's from the rate matrix **Q**. This amounts to using TPT directly to compute rates between "macrostates", i.e., chosen subsets of states. To prevent notational proliferation, let us consider in this section only a *general* jump process with rate matrix **K** with elements k_{ij} estimating the escape rate (probability per unit time)

from cell B_i to cell B_j, linking pairs of states indexed as $i = 1, ..., N$. Consider any macrostate α that is the union of some set of microstates. TPT permits the calculation of the rate between any pair of macrostates. To do that, let α be a reactant state, with all other macrostates, $\beta, \beta \neq \alpha$, as the aggregate product state. TPT gives the statistical properties of reactive trajectories for the reaction from reactant α to the product state. The essential quantities to compute are the forward committor function, q_i^+, the probability that the process starting at i will reach the product state before reaching α, and the backward committor function q_i^-, defined as the probability to last come from α rather than the product state arriving at i. When the process is time-reversible (i.e., the detailed balance is satisfied), $q_i^+ = 1 - q_i^-$. The discrete rate matrix ($\mathbf{Q}$) is used to calculate the forward committor function (q^+) through solving the constrained linear problem:

$$\begin{aligned} \mathbf{Q}q^+ &= 0 \\ s.t.\ q_i^+ &= 0 \ \ \forall\ i \in \alpha \\ q_i^+ &= 1 \ \ \forall\ i \in \beta \end{aligned} \tag{2.17}$$

Where α and β represent the reactant and product states respectively. This set of linear equations can be solved using standard numerical techniques [47]. The transition rate from α to β per unit time is then given by:

$$F_{\alpha,\beta} = \sum_{i \in \alpha, j \in \beta, i \neq j} \pi_i k_{ij} (1 - q_i^+), \tag{2.18}$$

where π_i is the equilibrium probability of state i. Thus, the normalized reaction rate is:

$$k_{\alpha,\beta} = \frac{F_{\alpha,\beta}}{\pi_\alpha} \tag{2.19}$$

where $\pi_\alpha = \sum_{i\in\alpha} \pi_i(1 - q_i^+)$. The total reaction rate from α to the product state is just:

$$k_{\alpha,product} = \sum_{\beta\neq\alpha} k_{\alpha,\beta} \tag{2.20}$$

Finally, the fraction of transition from α to β is:

$$f_{\alpha,\beta} = \frac{k_{\alpha,\beta}}{k_{\alpha,product}} \tag{2.21}$$

Using this formalism, we can compute rates connecting macrostates represented the active site and any solvent portal.

2.7 Steered Molecular Dynamics

In general, Steered Molecular Dynamics (SMD) [48] is a technique where a harmonic time-dependent potential is added to the standard Hamiltonian to facilitate unbinding of a ligand. SMD is typically applied when standard MD cannot yield ligand unbinding on practical time-scales. In our case, MD shows FOA stable in the active site for longer than 100 ns. Using SMD, FOA can be removed within several nanoseconds.

We originally planned to apply a variant of Markovian Milestoning in Voronoi Tesselations to MSOX wherein Steered Molecular Dynamics (SMD) would be used to generate the initial cells. This would eliminate the need for performing single-sweep and ZTS prior to application of MMVT, thus saving considerable time. The initial frames would be extracted from a long SMD trajectory where FOA was pulled out of the active site and into the solvent. MMVT would be run in each cell, and the cell center positions updated based on the cloud of hitting points after several nanoseconds. However, currently we are unable to prove that this approach would yield the correct pathways of greatest flux from the active site to the solvent.

Despite failing to prove that the correct pathways could be attained, FOA was successfully removed from the active site using SMD. Using the PLUMED [49] plugin integrated into NAMD 2.9 [38] we applied a constant force to FOA to remove it from the active site. Simulations were complete when FOA was completely solvated outside the substrate entryway. The direction of the force vector was determined by the center of mass of FOA initially, and a point in the solvent outside the substrate entryway where FOA was fully solvated. The resulting force vector was (0.000736 0.6742 0.7385). We then applied a constant force of 0.00005 Å/fs for 2 ns.

2.8 Trajectory Analysis

All analysis was performed with custom codes/scripts in C, TcL , FORTRAN, and Python. Figures were rendered in VMD [50] or created using gnuplot v4.2, GIMP v2.6, and XFIG v3.2. Samples of the NAMD configuration files used for MD, TAMD, force calculations, and milestoning can be found in Appendix A. Custom single sweep analysis codes for determining the RMSD of MSOX during equilibration, O_2 extraction, ensuring centers are 2.5 Å apart, as well as extraction of z and θ and calculation of the running average during force calculations are found in Appendix B. Centers deposition and free energy reconstruction were performed using codes developed for our previous work on Mb [30, 51]. Custom milestoning codes and scripts for generating the initial system configurations as well as analyzing the resulting data can be found in Appendix C. Determination of MFPTs was performed using a Python code developed for our study of CO migration pathways in Mb [2].

3. Aim I: Determination of Potential Oxygen Transport Pathways

3.1 Introduction to Aim I

Recent work has illustrated that small dissolved gases can access protein interiors through relatively small, fluctuating channels (e.g., the histidine gate in myoglobin [16, 17]). It is likely then that enzymes which utilize gas molecules as reactants will exhibit such channels. MSOX is no exception as oxygen must enter from the bulk solvent and travel to the active site which can be partially blocked when a ligand is present. However, experimental results suggest that sarcosine oxidation and oxygen reduction may occur on the *re*- and *si*-faces of the flavin ring, respectively [11, 12], which is consistent with the modified ping-pong mechanism (MPP). The MPP may be consistent with experimental data, but implies O_2 can oxidize the cofactor before the product imine is released [7].

If the modified ping pong mechanism is correct, it requires oxygen to enter and become oxidized while a substrate is present. Since the reduced enzyme can reduce molecular oxygen without needing a bound substrate, the dominance of the modified ping-pong mechanism suggests that the presence of a substrate in the active site somehow increases the rate at which the enzyme reduces oxygen. Our overarching hypothesis is that the substrate presence alters one or more pathways connecting solvent to the active site in such a way as to facilitate more rapid transport, relative to a substrate-free active site. Experimental results coupled with the crystal structure are not enough to determine these secondary passages to the active site, since the crystal structure only shows one clear route to the active site. We therefore turn to all atom simulations to investigate O_2 channels in MSOX, for which the composite method of Maragliano et al. [29] is well suited. In this approach, we first sample

the entire O_2 accessible interior volume of MSOX. From this set of points, we then extract a small number of locations at which to compute the mean force. These mean forces are the derivative of the free energy for O_2 at points throughout MSOX and into the nearby solvent. From here, the irregular mesh of points serve as input to a gradient-error minimizer that determines the optimal coefficients in a radial basis function expansion representing the free energy of O_2 as a function of position. As a point of reference, we take the free energy of O_2 to be zero in the solvent.

We then use this analytical free energy surface in two ways. First and most importantly, we can identify regions of low free energy through multiple walker steepest descent calculations. These regions are assumed to be indicative of where O_2 should localize. Knowing where oxygen localizes, we can then apply Zero Temperature String Method (ZTSM) to connect the locations of low free energy, and determine a minimum free energy pathway (MFEP) between any two wells. A MFEP is the "path of least resistance" to O_2 travel between any two sites on the analytical free energy surface. We assume these MFEP will carry the majority of the flux from the bulk solvent to the active site in the physical system as well. In this way we answer the following questions:

1. Where are the major sites for oxygen localization within MSOX?
2. What are the MFEPs connecting the solvent to the active site?
3. What differences if any exist between inhibitor bound and apo MSOX MFEPs?

3.2 Aim I Results

3.2.1 Molecular Dynamics

We first applied traditional molecular dynamics techniques to the apo and inhibitor bound systems. The goals were to achieve protein equilibration and assess

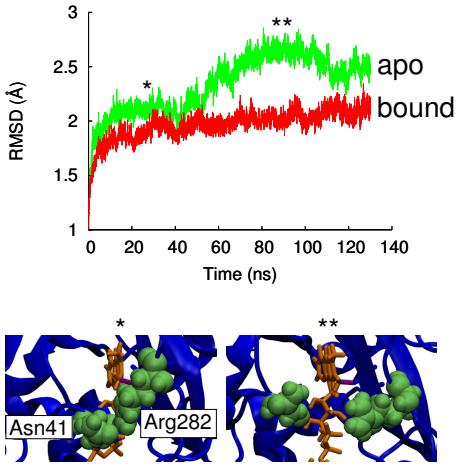

Figure 3.1: (Upper) C_α root-mean squared deviation (RMSD) from MD simulations of apo and FOA-bound MSOX. (Lower) Images from the apo simulations showing detail of the interaction between Asn41 and Arg282; here, "*" refers to the "closed-bridge" state and "**" to the "open-bridge" state.

whether standard techniques could provide sufficient interior sampling to answer the questions posed. Both apo and inhibitor bound systems were subjected to 130 ns of equilibration. Figure 3.1 shows root-mean squared deviation (RMSD) traces for each MD equilibration, showing both systems equilibrate after about 10 ns. The apo system undergoes slightly larger fluctuations than the inhibitor-bound system, which likely stems from the fact that the inhibitor's interactions with nearby residues in the active site can suppress fluctuations via steric crowding. As can be seen in the apo RMSD trace (Figure 3.1, green curve), a jump in RMSD occurs at about 50 ns. This corresponds to the spontaneous opening of the Asn41-Arg282 bridge of the flavin cleft, as illustrated by the renderings in the lower panels of Figure 3.1. The presence of this third long lived state necessitated the inclusion of a third system for single-sweep analysis. We therefore performed single sweep analysis on the following systems:

1. No inhibitor with an open cofactor admitting cleft. "apo-OB"

2. No inhibitor with a closed cofactor admitting cleft. “apo-CB”

3. Inhibitor present in the active site where the native substrate would bind. “bound”

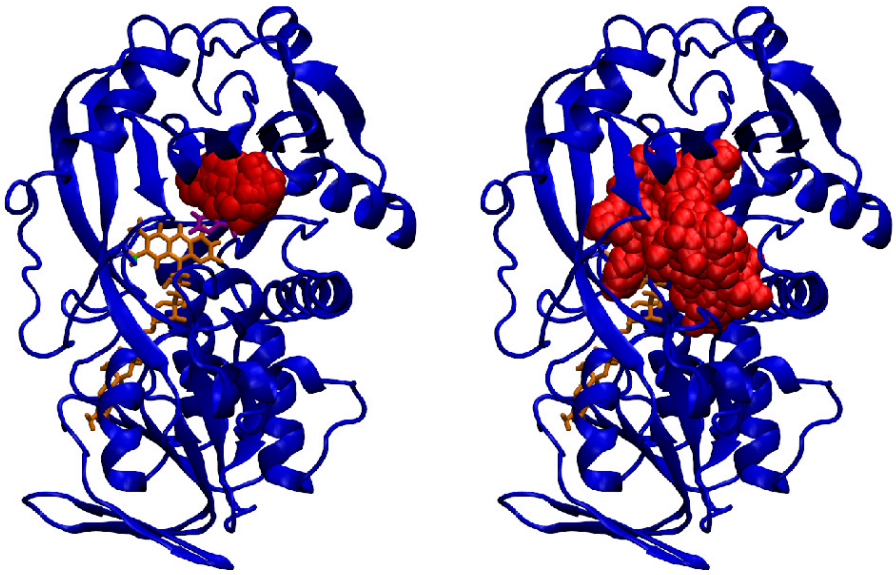

Figure 3.2: Cartoon structures of bound MSOX showing the location of sampled O_2 positions in red. (Left) 1 ns MD trajectory, and (right) 1 ns TAMD trajectory with $\bar{\beta}^{-1} = 3$ kcal/mol.

Under normal MD, O_2 sampling of the protein interior is severely limited on accessible computational time-scales. Figure 3.2 depicts sampling of O_2 locations from 1 ns MD and 1 ns TAMD simulations on bound MSOX. Both were initialized from the same O_2 location on the *re*-face of the flavin isoalloxazine ring. On comparable time scales, this clearly shows a lack of sufficient interior sampling from MD, thus requiring better sampling techniques.

3.2.2 Temperature Accelerated Molecular Dynamics

Single sweep requires sampling of as much of the entire interior of the system of interest as possible. Since MD cannot provide this sampling, we apply TAMD in the same way Maragliano et al. did to achieve complete interior sampling [29]. Briefly, TAMD accelerates the sampling of collective variables (CV's) $\boldsymbol{\theta}(\boldsymbol{x})$ in an MD simulation by tethering them to fictitiously hot auxiliary variables $\boldsymbol{z}$ with high friction $\overline{\gamma}$, such that the forces these variables experience approximate negative free-energy gradients on the free-energy surface of the CV's [26]. In essence, the often multidimensional CV of interest is allowed to evolve at a much higher temperature than the rest of the system so that it can overcome large free energy barriers and sample rare states. For MSOX, we chose our CV to be the center of mass of oxygen. Productive TAMD simulations where O_2 explored large portions of the interior volume prior to exiting were observed from 3-5 kcal/mol. As with MD, the active site loop (residues 55-60) remained closed for all simulations. Spontaneous closing or opening of the flavin admitting cleft was also not observed in either of the apo systems. To achieve sampling of the entire interior, we also needed to initialize TAMD simulations from each of four locations, two in each domain (Figure 3.3). For a complete description of TAMD parameters, see Methods 2.4.

3.2.3 Single Sweep

Single sweep can be applied once the interior of the system of interest has been sufficiently sampled. To achieve consistent sampling of the accessible interior volume of MSOX across the three systems of interest, we ran a total of 151 ns of TAMD sweeps (see Methods 2.4). For each system we then created a composite view of all TAMD frames with O_2 beginning inside MSOX and extending into the solvent until O_2 was completely solvated. This was typically several thousand frames. Figure 3.4

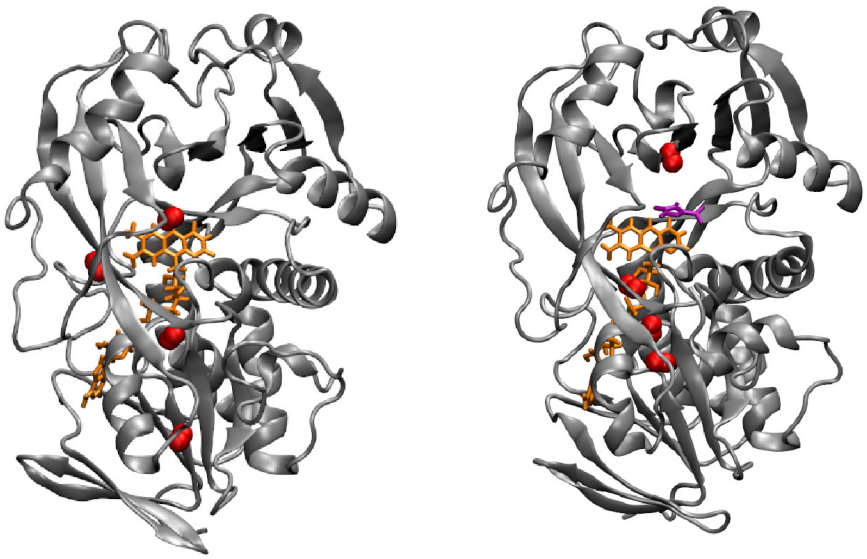

Figure 3.3: Cartoon structures of apo (left) and bound (right) MSOX showing the location of the four sites used for O_2 (red, space filling) initalization in TAMD. FADH (orange) and FOA (purple) are shown as tubes.

shows that TAMD is capable of sampling nearly the entire interior volume of MSOX in a short amount of time. Similar composites were generated for the apo-OB and bound systems as well. We then start with the first frame of the first simulation nearest the active site, and chose each frame that contained O_2 2.5 Å away from any other previously selected frame. This process is known as centers deposition, with each location for O_2 referred to as a "center". These are the points forming the irregular mesh on which mean forces are computed. A total of 1,033 centers were deposited across the three systems (see Methods 2.5). Figure 3.5 illustrates the irregular mesh of points generated for each of the three systems explored with single sweep. Despite nearly a factor of two difference in the number of centers deposited between apo and inhibitor bound MSOX, interior sampling remains nearly the same. The vast majority of additional centers occupy the solvent outside the protein but within the 10 Å cutoff distance for interatomic interactions. As will be shown later, these centers do not impact the pathways within MSOX. They are included to smooth

the analytical free energy surface outside the protein. Note that the volume spanned by the set of centers includes many important residues, including the active site loop (residues 55-60), Lys265, Phe256, Arg49, Cys315, and both the *re*- and *si*-faces of the flavin ring [9–11, 13, 14, 52].

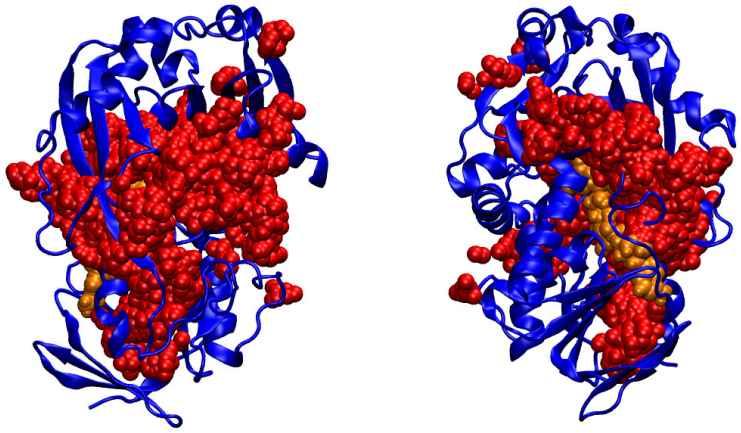

Figure 3.4: Composite TAMD trajectory for apo-CB MSOX illustrating accessible interior sampling via TAMD. A total of 7,456 frames totaling 36 ns are shown. The right panel is rotated 180 degrees to illustrate sampling in the cofactor admitting cleft and on the *si*-face

With centers deposited we can now determine the mean force acting on each center by applying the TAMD protocol with infinite friction. This prevents translational motion of the O_2 molecule, yet allows nearby forces to build up on its center of mass. Mean forces were calculated after the average of each individual x,y, and z component of the force acting on O_2 at the location saturated. For each of the 1,033 centers, a plot was generated, showing the running average of the forces (Figure 3.6). After each component of the force saturates, the value of the final running average is the

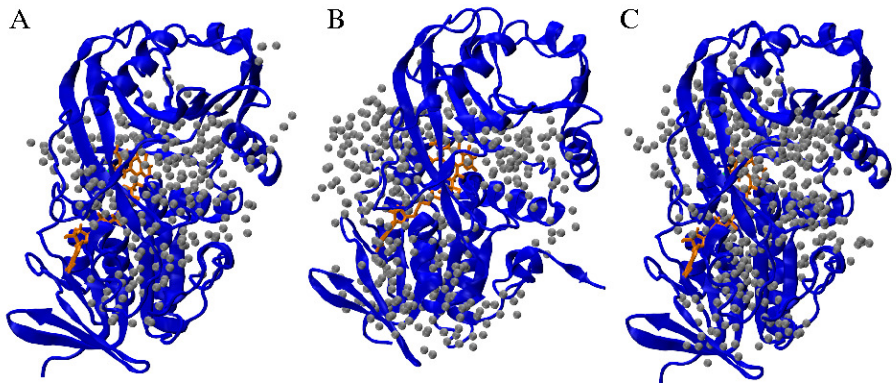

Figure 3.5: Cartoon structures of MSOX showing location of centers harvested from (A) apo-CB MSOX, (B) apo-OB MSOX, and (C) bound TAMD sweeps.

force vector acting on that location. This step is the one of the most computationally expensive, requiring approximately eight hours per *successful* force saturation on two nodes (16 processors each) at a time, totaling approximately 250,000 CPU hours. Utilizing the Stampede supercomputer at the Texas Advanced Computing Center (part of the XSEDE consortium [53]), up to fifty mean force calculations were performed at once. While most simulations could have been performed locally, this step could not as it would have taken twice as long per calculation and only one could be performed at a time, thus requiring several years to complete.

After completing the mean force calculations, the analytical free energy is reconstructed using a radial basis function (RBF) expansion with the Gaussian RBF (see Methods 2.5). The next step is to apply multiple walker steepest descent minimization to determine regions of low free energy. These areas are referred to as local minima. For each center previously identified, a walker is allowed to "fall" into the nearest free energy minimum via free energy gradient minimization on the analytical surface. From the final walker locations, sets of interesting local minima are identified. The deepest minima calculated were on the order of -10 kcal/mol, relative to O_2 in solvent. The deepest minima identified were observed to occur at the same locations

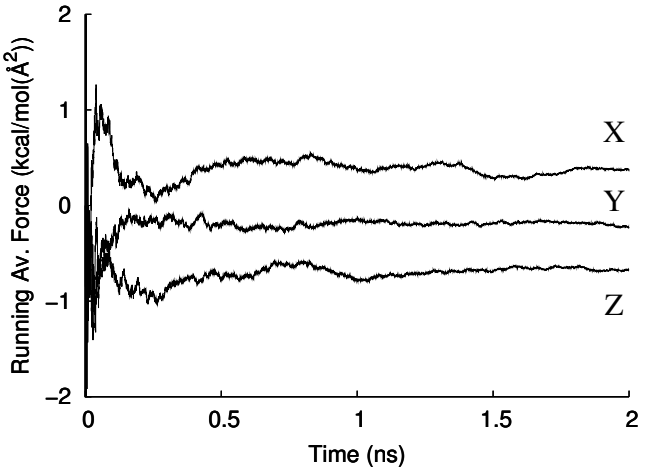

Figure 3.6: Running average forces for one representative center.

for all three systems. Fifty-one unique minima were identified for the apo-CB system, 50 for apo-OB, and 96 for bound MSOX. The bound system was observed to display many more shallow local minima which, as will be shown, had very little effect on the free-energy profiles along the pathways of minimal free energy. Figure 3.7 illustrates the locations and depth for each of the local minima identified via single sweep.

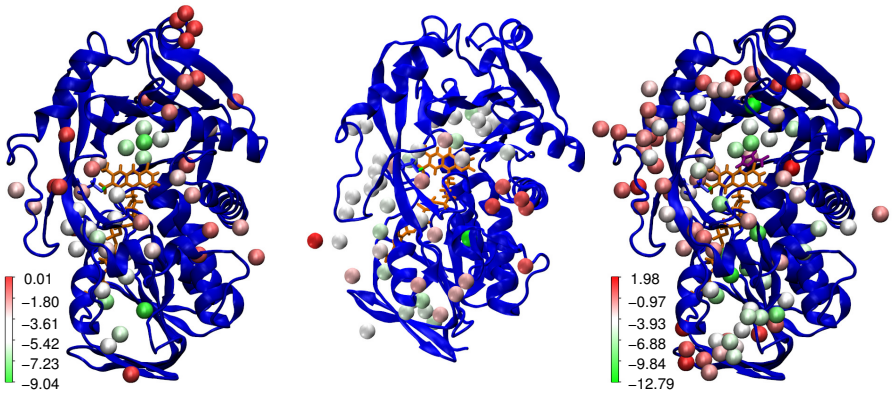

Figure 3.7: The local minima identified via single sweep for apo-CB (left), apo-OB (middle), and bound (right) MSOX. The scale (kcal/mol) for both apo (left,middle) is the same. The protein (blue) is shown in a cartoon representation, with FADH (orange) and FOA (purple) in tube.

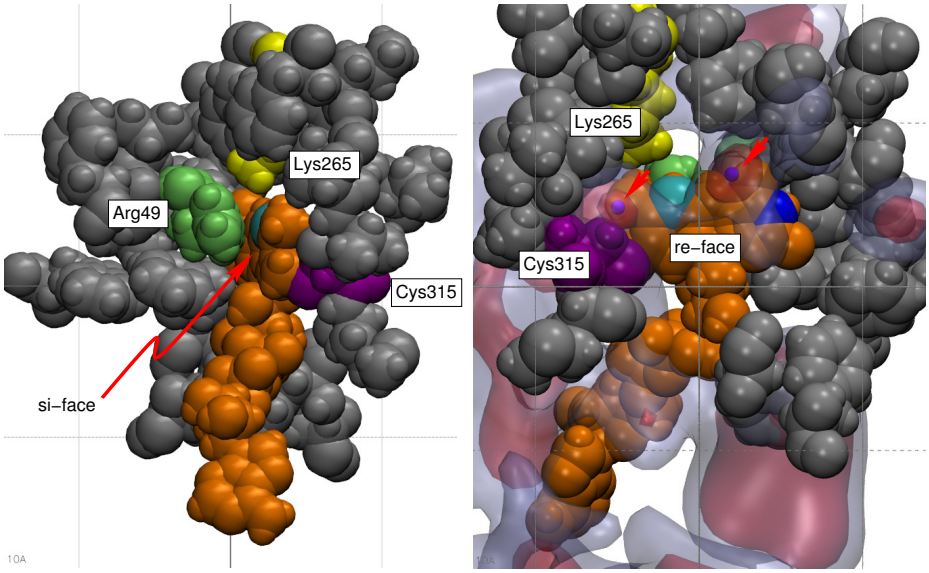

Figure 3.8: (Left) Space-filling view of the flavin co-factor (orange) and all atoms in hydrophobic residues within 10 Å of the flavin atoms C(6) and N(3). The crowded *si*-face has little to no room for molecular oxygen to interact with the isoalloxazine ring. (right) Rotated view of same set of atoms as in the left panel, showing the *re*-face of the isoalloxazine ring. Superimposed on this view are isosurfaces of the free energy at -5 kcal/mol (red) and -3 kcal/mol (light blue). Two local free-energy minima on the *re*-face are indicated with red arrows. Flavin atoms C(6) is shown in cyan and N(3) in blue. A 10 Å grid is overlaid on the right panel.

3.2.4 Minimum Free Energy Pathways

To find pathways of minimum free energy between any two of these minima, the zero-temperature string method was applied [35]. Briefly, for two minima A and B, a line segment connecting them is discretized into N sites, each of which is a walker that is allowed to move according to the local gradient in free energy subject to a reparameterization step that keeps the site-site separation distance along the string uniform. The string of sites thus traces the path of least resistance connecting the two minima. String convergence for this investigation was achieved when the change in free energy between successive calculations was approximately 10^{-5} kcal/mol. We typically used $N = 50$ discretization points on each strings.

Each of the three systems displayed the same two distinct minima on the *re*-face of the flavin ring. One occurs near flavin atom C(6) and the other near N(3) (see Figure 3.8). These two minima are features of a larger hydrophobic basin surrounding the *re*-face. It is important to note that no such minimum occurs on the *si*-face. Lysine residues near the flavin ring have been shown to be important catalytically not only in MSOX, but histone demethylases LSD1 and LSD2 as well as other systems [9, 21, 31, 54]. Lys265 is directly above C(6), and a local minimum nearby further supports the idea that oxygen activation occurs near it. However, since both minima are located on the *re*-face of the ring, we posit that at least the initial interaction of neutral molecular oxygen with the isoalloxazine ring is a metastable encounter complex with the O_2 on the *re*-face. It may be that the first electron transfer event creates a superoxide anion and the one-electron reduced flavin radical such that the superoxide transits to the end of the Lys265 side-chain to the *si*-face, but this would require relative motion of the flavin and Arg49 to make room. However, considering the inhibitor already serves to suppress motion of nearby residues in the active site, this seems unlikely. It is more likely that the ligand suppresses motion of nearby residues to help form a stable site for the formation of the superoxide so that it does not react away too quickly. The limitation of only considering molecular oxygen in our simulations means we cannot directly address the hypothesis that activation and sarcosine oxidation may occur on opposite faces of the flavin ring, as suggested by Zhao et al. [9]. Both minima observed on the *re*-face of the flavin ring are not shallow, approximately 6-7 kcal/mol, indicating these are stable sites for O_2 to localize. However, based on the location of the substrate-mimicking inhibitor and the presence of two oxygen minima nearby in the apo systems, this location may also simply be the sarcosine activation site, consistent with experimental results [6].

Returning to the local minima, for each minimum near the surface ZTSM located

a MFEP to the active site. The active site is defined as the minimum near N(3) on the *re*-face of the flavin ring. It is approximately where the substrate-mimicking inhibitor lies. This resulted in four distinct pathways, designated I-IV. With the exception of pathway I, each displays a high degree of geometrical similarity among the three systems. System-specific pathways are referred to using prefixes "apo-CB", "apo-OB", or "bound". The pathways are represented as tubes in Figure 3.9. We stress that these pathways are assumed to carry the majority of the O_2 flux from the solvent to the active site.

Pathway I enters MSOX between the loop connecting βF2 with αC1 (residues 32-59) and the loop connecting βC7 with αC4 (residues 268-289). These loops form part of the cleft that admits FADH into MSOX, and are seen to open and close in a gate-like fashion in the standard MD simulations. Inside MSOX, pathway I passes next to the catalytically active Lys265. Both apo-CB-I and apo-OB-I enter MSOX this way. Apo-CB-I then passes over the *si*-face of the flavin ring and directly under Lys265 when viewing the flavin's *si*-face before terminating at the active site. It must be stressed that while apo-CB-I passes through what would be the oxygen activation site reported in the literature [9], no free energy minimum is observed there. Apo-OB-I goes over the Cys315 linkage before passing through both minima on the *re*-face and terminating at the active site. The bound-I string however, enters MSOX above the entrance loops. It then passes between Arg49 and Lys265, crossing above the *si*-face and terminating at the active site. Consequently, bound-I passes the loop connecting βC6 to βC7 (residues 256-264). While apo-OB-I passes through the two distinct minima on the *re*-face of the flavin ring, apo-CB-I and bound-I avoid the minimum near C(6).

Pathways apo-OB-II, apo-CB-II, and bound-II pass through the middle of the cofactor admitting gate defined by Asn42 and Arg282. Oxygen then travels along

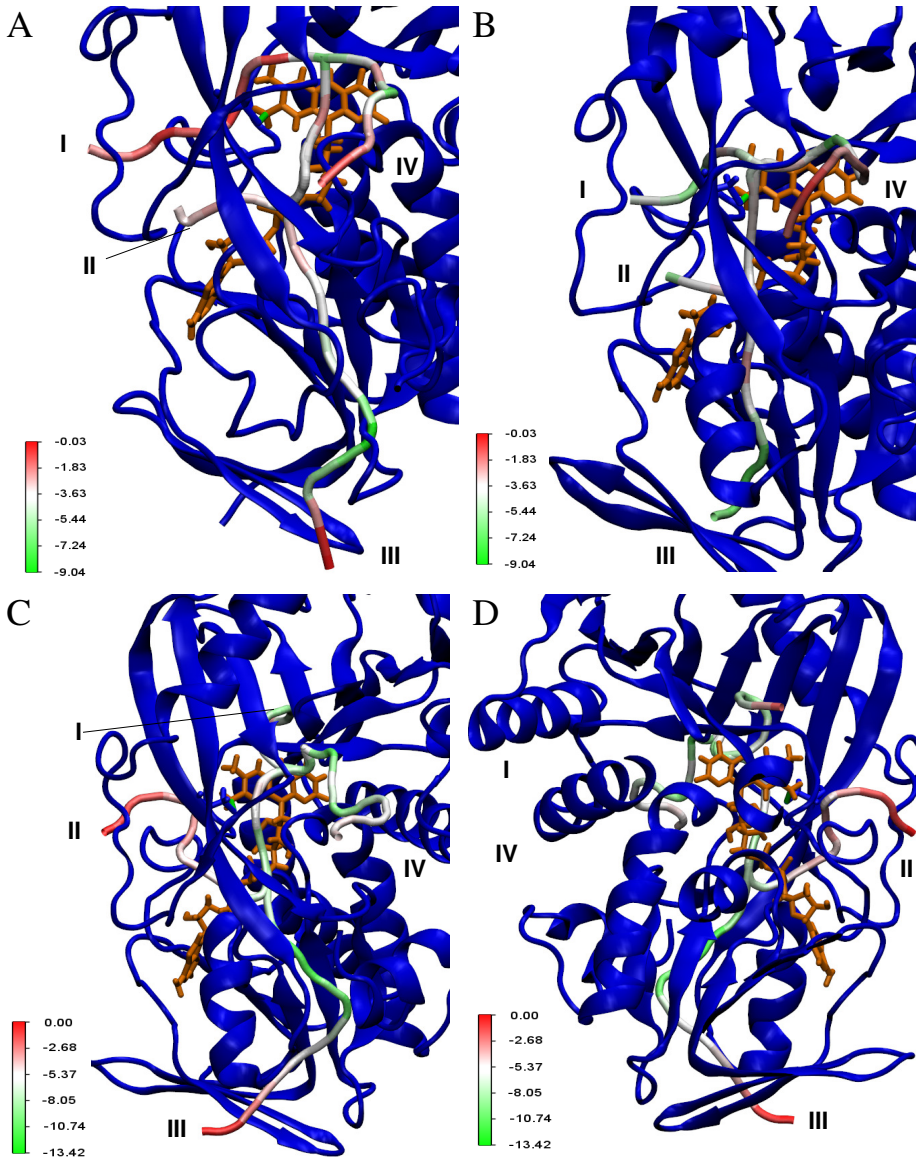

Figure 3.9: (A) Pathways apo-CB-I, -II, -III, and -IV overlaid on the MSOX structure. (B) Pathways apo-OB-I, -II, -III, and -IV (C) Pathways bound-I, -II, -III and -IV. (D) Rotated view (180° along of bound pathways to provide a better view of pathways bound-II and -IV. Free energy along each pathway is indicated by the pathway color.

the sugar backbone of FADH, entering the binding pocket after passing between the *re*-face of the flavin ring and the loop connecting βF9 with αF4 (residues 340-350). All three paths are geometrically nearly identical and pass through the two distinct minima identified on the *re*-face.

Pathways apo-OB-III, apo-CB-III, and bound-III traverse the entire FAD binding domain and most of the catalytic domain to arrive at the active site. The path begins at an entry portal far from the site, bounded by two loops: the first loop connects βF8 to βF9 (residues 330-335) and the second αF3 to βC4 (residues 212-219). The pathways continue past a loop connecting βF7 to αF3 (residues 199-205). Subsequently, oxygen travels along the sugar backbone of the FADH, converging with pathway II near the flavin ring. As with II, pathways apo-OB-III, apo-CB-III, and bound-III are geometrically nearly identical and pass through the two distinct minima identified on the *re*-face.

Pathways apo-OB-IV, apo-CB-IV, and bound-IV enter MSOX via passage through the large opening identified as the sarcosine entryway [6]. All three pass between the closed active site loop (residues 55-60) and residues 268-272, part of the loop connecting βC7 with αC4, en route to the sarcosine activation site. Again, all type-IV paths are geometrically nearly identical across the three systems. However, since they enter through the substrate access channel and terminate directly on the minimum near N(6), pathway IV does not pass through the minimum near C(6).

We reiterate that none of the pathways pass through a minimum on the *si*-face of the flavin ring before terminating at the sarcosine activation site. Lack of a minimum on the *si*-face of the flavin ring is a concern because as mentioned previously, it has been suggested as the site of oxygen activation [9]. The TAMD trajectories from which centers were harvested sample this region, and several centers at which mean forces were calculated were located on the *si*-face, so if a minimum did exist it would have been detected. To double-check whether such a minimum would occur, an additional center was added on the *si*-face under Lys265 and its mean force was evaluated. The PMF was reconstructed again for the apo-CB system, and no additional minimum was observed. The bound system was also observed to contain no additional minimum.

However, the minimum near C(6) on the *re*-face of the flavin ring is conserved in all three cases. This suggests that the site may be important to catalysis and possibly be the O_2 activation site.

The area within 10 Å (the electrostatic cutoff distance) of C(6) encompasses both *re*-face minima, and exhibits interesting characteristics (see Figure 3.8). Although the region around the flavin ring is reported as basic, basic residues primarily occur on the *si*-face. The *re*-face exhibits mostly non-polar residues, a feature which would stabilize the superoxide. The presence of multiple non-polar residues is significant. Recent work has suggested that oxygen activation sites in flavoenzymes require a non-polar residue nearby to aid in desolvation, optimize site geometry, and maximize electrostatic effects on molecular oxygen. It was therefore suggested that Phe256 in MSOX would be the essential non-polar residue owing to its proximity to Lys265 [14]. Phe256 is well within the cutoff for interactions with oxygen located at the C(6) minimum. The minima near C(6) therefore exhibits nearly ideal conditions for oxygen activation: proximity to the catalytic Lys265, multiple nearby non-polar residues to stabilize the site, and is easily accessed by oxygen in all simulations.

4. Aim II: Refinement of the Kinetic Network with Markovian Milestoning and Comparison to Experimentally Determined Rate Constants

4.1 Introduction to Aim II

Aim I provided several key pieces of information for analysis of O_2 transport within MSOX. We first determined the free energy of O_2 as a function of position within the protein. Using the free energy reconstruction, we located many local minima for the three systems in question (Apo-OB, Apo-CB, and Bound), along with the MFEPs connecting surface minima to the active site. This yielded four primary MFEPs (Figure 3.9) as well as a potential oxygen activation site on the *re*-face of the flavin isoalloxazine ring. Recall that our ultimate goal is to calculate one or more quantities which can be compared to experiment in order to predict whether entry or exit of O_2 is rate limiting, and to assess whether modified ping-pong (MPP) or ping-pong (PP) is the more likely mechanism.

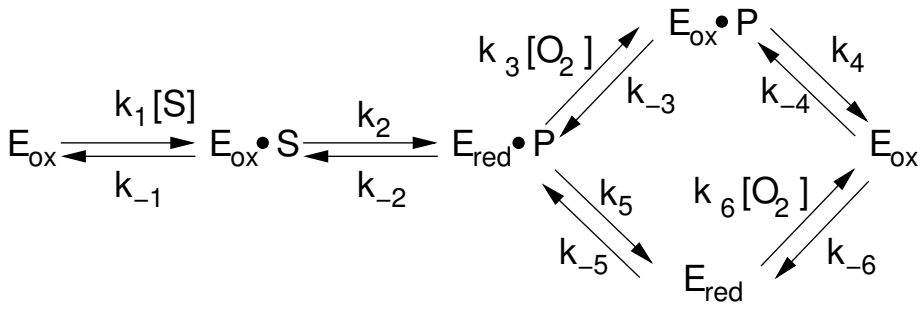

Figure 4.1: Elementary mechanisms of the MSOX catalytic cycle [1]. The lower branch represents the classical ping-pong mechanism, while the upper branch represents the modified ping-pong mechanism. Note that in each mechanism, the pseudo-first order rate constants $k_3[O_2]$ (modified) and $k_6[O_2]$ (classical) represent both entry of O_2 and oxidation of the enzyme's flavin.

We begin by examining the elementary mechanisms of the MSOX catalytic cycle (Figure 4.1). Of the two competing theories, the primary difference between MPP and PP is the order by which oxidation of the flavin cofactor and product release occur. If the cofactor is oxidized first, and then the substrate binds for the reaction to occur, PP is the more likely mechanism (Figure 4.1 top branch). If oxidation of the cofactor can occur while a substrate (or product) is present, then MPP is more likely (Figure 4.1 bottom branch). Modeling entry of reactants, followed by the reaction and release of products is outside the scope of what we can do with the systems as constructed. It would require, at a minimum, a force field which can model polarization and charge transfer interactions leading to the superoxide as well as single sweep analysis for each component. We therefore look at what can be observed with our model.

We can ask how does the motion of O_2 within MSOX depend on the presence or absence of the substrate-mimicking inhibitor FOA. It would be possible for us to compute both k_3 and k_6 from Figure 4.1 if the rate of reaction were fast compared to the rate of entry; we don't know this to be the case. We cannot model oxidation of the flavin since we have a non-reactive force field. Consequently, any computed rate constants should be faster than experiment, since the combination of entry, reaction, and exit (which is what the experimental results provide) would be slower than any one step individually. So, given that experimental values of k_3 and k_6 exist, our rate constants for entry can be used to assess whether or not entry limits the rate of O_2 reduction. Combined with our predicted exit rates, we can judge whether or not the substrate influences the average residence time of O_2 in the active site.

The question now becomes: "How do I calculate whether entry or exit of O_2 is potentially rate limiting?" Most previous molecular simulation work has focused on identifying possible routes O_2 takes between the bulk solvent and the active site in similar enzymes without specific consideration of rates [25, 31–33]. Although these

works support the idea that small gas molecules can access buried sites by multiple pathways, it remains challenging to determine which pathways contribute the most to the rate at which O_2 accesses an active site, since none provide a direct way to calculate such rates.

Markovian Milestoning develops a Markov state model describing the rates of transitions between important states. It is similar in spirit to using Brownian dynamics and a reduced system description to develop a state model describing transitions. In fact milestoning has been shown to yield comparable results to both brute force MD and Brownian dynamics to develop Markov state models [55, 56]. Our group recently adapted the MD-based method of Markovian Milestoning [34] to compute rates of diffusion, entry, and exit of small molecules in proteins, utilizing it to study CO entry and exit from myoglobin [2]. Yu et al. identified a full kinetic network for CO in myoglobin, and determined which of the many interconnected routes dominated transport [2]. Using that framework, we apply and adapt Markovian Milestoning to MSOX. Our work takes milestoning one step further as we now look to examine multiple states of MSOX. In doing so, we are able to offer diffusion-based evidence in support of the modified ping-pong mechanism, which is also supported experimentally. With that in hand we can then combine structural analysis of the identified transport routes with their associated entry and exit times to predict whether entry or exit is rate limiting.

We applied a version of milestoning referred to as Markovian Milestoning in Voronoi Tesselations (MMVT) [34]. The general idea is to take a reactive trajectory and break it into extremely small sections and generate transition statistics before applying transition path theory (TPT) to develop a Markov state model in which the states are the section boundaries. Within each section, we generate statistics on how long it takes the molecule of interest (in this case O_2) to transition forwards or

backwards. This is different from the original formulation of milestoning in the sense we do not need to reinitialize from a first hitting point distribution [41]. Instead, using a Voronoi tesselation, we can watch statistics within each cell saturate from long restrained MD. We then use the statistics generated to for solve the committor function which describes the probability of a reactive trajectory reaching the product state before the reactant. The end result is Mean First Passage Times (MFPTs) describing entry and exit for O_2, which can be processed to approximate second order rate constants for entry and first order rate constants for exit. Using MMVT, we therefore hope to answer:

1. What are the MFPTs for each MFEP, and which dominate O_2 transport?
2. What are the overall MFPTs describing entry and exit for O_2?
3. Which of the two competing mechanisms (MPP or PP) is supported by our results?
4. Based on the overall MFPTs, what are the approximate values for k_{entry} and k_{exit}, and does entry or exit limit the overall reaction?

4.2 Aim II Results

4.2.1 Analysis of the MFEPs from Single Sweep

Determination of entry and exit rates begins with analysis of the MFEPs discovered from single sweep. Generally, we observe pathways in the ligand-bound system to have lower free energy than their counterparts in the apo-CB or apo-OB systems. A lower free energy indicates a more stable site for oxygen to remain, hence a longer time to leave that spot. This should inhibit transport to the activation site in the bound system as oxygen will tend to remain in the deepest minima. In contrast, with ligand unbound, the deep local minima along the pathways attenuate somewhat, which

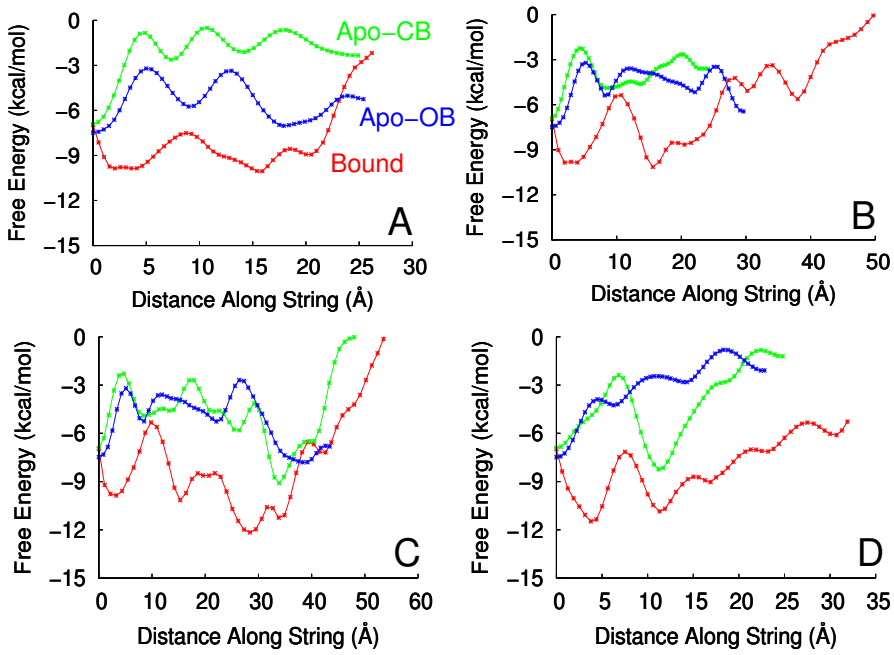

Figure 4.2: Free energy as a function of distance along each pathway from the activation site for pathways (A) apo-CB-I, apo-OB-I and bound-I; (B) apo-CB-II, apo-OB-II and bound-II; (C) apo-CB-III, apo-OB-III, and bound-III; and (D) apo-CB-IV, apo-OB-IV and bound-IV. Apo-CB is show in green while apo-OB is shown in blue and bound in red. All graphs begin with oxygen in the active site and end at the nearest minimum to the protein surface where interactions with the solvent occur. Oxygen in solution has a reference value of 0 kcal/mol.

would likely promote transport. Consequently, we expect entry and exit rates in the bound system to be slower than in apo. It is not surprising that overall geometrical similarities in pathways exist since the primary difference among the three systems is presence of an inhibitor. However, the large differences in free energies along these pathways between apo and ligand-bound MSOX are unexpected. Evidently, ligand binding and unbinding must cause subtle conformational changes or changes in local fluctuations which affect oxygen transport. A deeper analysis of the residues along each pathway reveals many of the bases responsible for these differences.

Apo-CB-I and apo-OB-I appear to be the only MFEPs affected by cleft opening and closing. When closed, the flavin ring shifts approximately 2 Å in the positive direction normal to the *re*-face, helping to expose the *si*-face. This is a feature conserved

in bound-I. Additionally, residues 42−44 are oriented under the *si*-face. Apo-OB-I in contrast has a *si*-face that is partially blocked by residues 42−44. This slight shift of the flavin ring caused by cleft motion is likely the cause of both the free energy and geometric differences between apo-OB-I and apo-CB-I (the Lys pathway). It is interesting to note that when examining bound-I, there were MFEPs which caused it to be identical to bound-II (Gate). Since the bound system contains a closed bridge, a MFEP similar to apo-CB-I was sought. To have bound-I cross the *si*-face required usage of a different starting point. However, it still passed Lys265 approximately where apo-CB-I did. Incidentally, bound-I then failed to pass through the minimum near C(6) on the *re*-face. Ligand binding may therefore cause the route previously taken by apo-CB-I and apo-OB-I to be adversely affected due to crowding on the *re*-face.

Access to and from pathways apo-II and bound-II (Gate) is largely controlled by loops 268−289 and 32−45. We initially hypothesized that opening and closing of the cleft would strongly influence the free energy near the end of the MFEP. However, examination of the free energy profiles indicates that opening and closing of the cleft does not influence the energetics significantly. The largest free energy differences are within the cutoff distance (10 Å) of the inhibitor. There are also no other major structural differences between the apo and bound states near the MFEPs. Thus, inhibitor binding plays a larger role in influencing energetics along pathway II than does disposition of the cleft.

Access to apo-CB-III, apo-OB-III, and bound-III (Lower) is controlled by loops 212−219, 330−335 and βF4. Apo-CB-III and apo-OB-III share similar energetics, and structurally are nearly identical within the vicinity of the MFEP. When compared to bound-III, several subtle differences become apparent. One of the controlling loops, 212−219, rotates inward, apparently trapping oxygen. This may be the cause of

the deep minima seen approximately 20 Å along the string. Ligand unbinding is correlated with relaxation of the loop, opening the cavity and allowing oxygen to escape. Rotation of this loop may not be caused by ligand binding however, as it is within the cutoff for interactions with residues 375−385. Residues 375-385 include the C-terminus and are completely solvent exposed. They fluctuate greatly over the course of any simulation and interact with loop 212−219. It is more likely that these interactions are the primary cause of loop shifts resulting in deeper minima. Path III for all three systems is the longest of the four identified.

Path IV (Sarc), which traverses the substrate entryway, bears three MFEPs which are nearly identical. For both apo and ligand-bound MSOX, the active site loop (residues 55−60) remains in the closed configuration for the duration of all simulations. When comparing apo-CB-IV and apo-OB-IV, the energetics are similar except when the MFEP is near Glu57. Apo-CB-IV features Glu57 pointed inward toward the flavin ring. This places it nearly in van der Waals contact with the MFEP at the deepest minima near 10 Å. Apo-OB-IV contains Glu57 pointed out toward the solvent. As such, it is further from the MFEP. This may not be the sole cause for the large discrepancy in free energy between the MFEPs, but it is the only major structural difference between the two in the region of the MFEP. When compared to bound-IV, there are several differences which must be noted. Tyr55 points away from the channel when bound, yet towards the channel in the apo systems. Glu57 is pointed toward the flavin ring similar to apo-CB-IV. This makes sense because the free energy near Glu57 for apo-CB-IV and bound-IV is similar in that region. It also reinforces the observation that orientation of Glu57 affects free energy along the substrate entryway. The final major difference is binding of the inhibitor. We observe the largest free energy differences to occur within the cutoff distance (10 Å) for non-bonded interactions with the inhibitor. Therefore, interactions with the inhibitor, as

well as shifts in Tyr55 and Glu57, result in a path of significantly lower free energy.

The sensitivity observed in the O_2 pathway thermodynamics to ligand binding suggests a possible link to the observation that high product concentrations enhance flavoenzyme kinetics [14]. One interpretation is that positively charged products play the role of stabilizing superoxide anion for certain flavoenzymes that, unlike MSOX, lack positively charged side-chains in the vicinity of the flavin ring. Our results hint at another possibility, at least for MSOX. When MSOX is substrate-bound, many deep local minima for O_2 along channels connecting solvent to the *re*-face cavity help it absorb O_2 from solution. If they were static features of the protein structure, however, these minima would presumably be detrimental for the processing of O_2 at the flavin ring. We see they are not static, but attenuate significantly in apo-MSOX, meaning that the channels should easily provide access for O_2 to the substrate/product-free flavin ring. A higher substrate concentration would lengthen the time MSOX spends in a substrate-bound state, sponging O_2 from the surrounding solution so that, once product release is initiated, a nearby O_2 can readily access the flavin ring and begin the next catalytic cycle. We were able to identify the structural shifts in backbone segments and sidechains that accompany ligand binding to explain how the depths of these minima are allosterically modulated. It is worth pointing out that the ILS simulations of O_2 in 12/15-lipoxygenase by Saam et al. [25] show that binding the ligand arachidonic acid closes one O_2 diffusion channel but opens another. Our work on MSOX shows that ligand binding can act more subtly by altering pathway thermodynamics without completely shutting them off.

It is not clear from this work whether or not MSOX is special in displaying ligand-dependent "cryptic" allosteric sites for O_2. However, the ability of proteins to display otherwise hidden secondary binding sites upon binding of a primary partner molecule is certainly gaining attention. For example, recent analysis of long all-atom MD

simulations by Bowman and Geissler clearly illustrate the existence of cryptic sites for small molecules in a variety of proteins [57]. Our work suggests the single-sweep approach is a viable choice of methods for discovering such sites.

4.2.2 MMVT Setup

The free energy profiles for the MFEPs identified with single sweep illustrate how binding of a ligand can have subtle yet far reaching effects on how easily O_2 accesses the active site. Based on the local structure surrounding the MFEPs and their free energy profiles, we expect rate constants for bound MSOX to be slower than apo. While informative, this does not provide adequate results to comment on whether entry or exit is rate limiting, and if MPP or PP is more likely. Much like work on other proteins, we have only uncovered potential transport routes, although in much greater detail. Thus, we consider the network of connecting MFEPs to be only an estimate of the true kinetic network present in MSOX. We now apply Markovian Milestoning in Voronoi Tesselations (MMVT) to compute the kinetic network and assess entry and exit rates.

Use of a Voronoi tesselation requires space to be discretized into meaningful locations for the cell centers. Since we assume the MFEP carries the majority of the flux for O_2, a logical choice for the centers of the tesselation is the discretized MFEP. Each of the $50-80$ images along an individual MFEP represents the center of a cell within the tesselation. We then run restrained MD inside each cell, and generate transition statistics to build a continuous time Markov jump process.

To begin MMVT, we extract the location of each image along each MFEP for all three systems. This resulted in approximately 690 cell center locations. Then, using the composite TAMD trajectories from single sweep, we select the frame closest to the individual cell center. To keep all restrained MD consistent, the O_2 present in

the system for that frame is moved precisely to the location of cell center that was extracted. It is important to note that this step is not necessary. The initial position within a cell is arbitrary as we are examining saturation of each $k_{i \to j}$ within a given cell. If the transitions stay centered about the MFEP, a poor starting position would not influence the statistics significantly, since we typically observed thousands to tens of thousands of transitions per cell. We chose to initialize at the center of each cell to keep all starting points uniform and easily reproduced. Thus, a total of 690 individual cells were initially generated.

To run MMVT in NAMD [38] we needed to implement a way to keep the O_2 confined to an invidual Voronoi cell. To achieve this, our group implemented a patch to NAMD 2.9 which, upon violation of a Voronoi boundary, rewinds all atom velocities by one timestep, then negates them and rewinds the positions of all atoms by one timestep [2].

4.2.3 Regarding Simulation Stability and Sampling

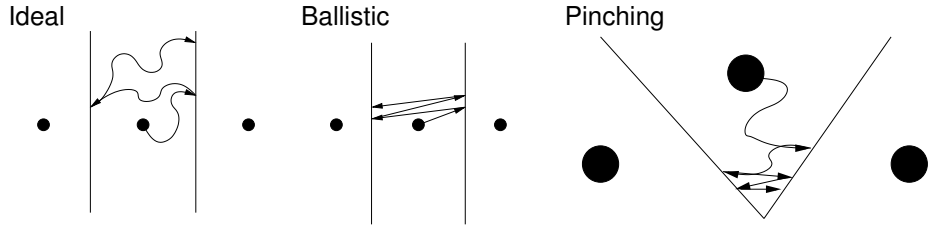

Figure 4.3: Cartoon examples of an ideal, ballistic, and pinching trajectory. Notice that the ideal trajectory samples large regions of the cell prior to hitting another boundary. In the ballistic case, motion of the O_2 follows a ping-pong like motion back and forth. For pinching, the trajectory begins ideal, but quickly deteriorates at the V into extreme ballistic motion.

Two cases exist where suspect data can be obtained from MMVT. The first occurs when the cell centers are too close to each other. If the centers are too close, it is

possible for ballistic motion to occur wherein O_2 transits between cell boundaries too fast, resulting in unrealistic transition times. From our work on Mb, we determined that a good starting point is to have each center at least 0.667 Å away from each other. Therefore, in MSOX, we attempted to initialize the kinetic network with each Voronoi center approximately 0.667 Å away from each other. Unfortunately, ballistic motion does not necessarily cause instability, and is often discovered after the fact upon analysis of the transition data.

The second case is more extreme and presents itself while performing the individual restrained MD simulations. It is another inherent limitation to the discretization of space. Many of the MFEPs snake through MSOX, curving around the interior to eventually reach the active site. If we visualize the Voronoi tesselation formed by these snaking MFEPs, we notice that there are multiple regions where the cell boundaries form a V-like shape (Figure 4.3 right). In these regions, we experienced a great deal of simulation instability, resulting in signficant losses to time, as each successful segment of a simulation requires up to eight hours of real time to complete on the TACC supercomputer Stampede using two nodes (16 processors each). Some cells were run on the Drexel supercomputer Proteus, but required $10-12$ hours per segment on two nodes. Multiple cells required up to eight segments (64 real hours), and several as many as ten (80 hours) to achieve saturation. Incomplete segments needed to be discarded and redone as the statistics provided were inaccurate. Additionally, analysis of the kinetic network could not be performed until *all* cells were completed.

Close examination of why instability occurs pointed directly to the highly curved regions of each MFEP. Although we still assume the MFEP carries the majority of the flux for O_2 in MSOX, we discovered that it is possible for O_2 to traverse a slightly different path. MFEPs only determine the path of least resistance between

sets of minima. They do not describe the surrounding free energy landscape or the height of any barriers which may separate portions of the MFEP that loop near each other. Therefore, it is possible that very small free energy barriers exist which can be overcome via normal thermal fluctuations. Essentially, the MFEP is a 1-D curve that tries to denote the effective tube-like shape that defines the region of maximal O_2 flux, and as such, it cannot represent the width or boundaries of that tube. Fine, high-curvature features of an MFEP are not necessarily representable by a finite-width tube.

If these small barriers exist near highly curved regions, the neighboring cells form a natural pinch point in the tesselation. As a result, O_2 will attempt to transition near the pinch point, resulting in extreme ballistic motion. Extreme ballistic motion is defined to be transitions which occur on the order of one timestep ($1-2$ fs). Recall that at each attempted transition, our patch instructs NAMD to reverse the velocity of O_2 and rewind the position for every atom within the system. If this occurs frequently over a short period of time, it can result in improper bond lengths, angles, and orientations. Therefore, transitions on the order of one timestep can cause the simulation to crash rendering all data in that segment useless.

To combat the instability resulting from "pinching" and any other discretization complications, we chose to combine cells exhibiting this behavior. Any cell which crashed three or more times was immediately merged with the nearest completed cell. In this way, the V-like shape formed by sharply curved regions of space becomes more U-like. We continued to combine cells until a stable simulation was achieved. Unfortunately, a consequence of merging cells is that nearby neighbors which could transition into the now missing cell needed to be redone as the tesselation itself has changed. In this way, we iterated over the kinetic network until all cells were stable and saturated.

4.2.4 Kinetic Network Refinement and Solvent Portals

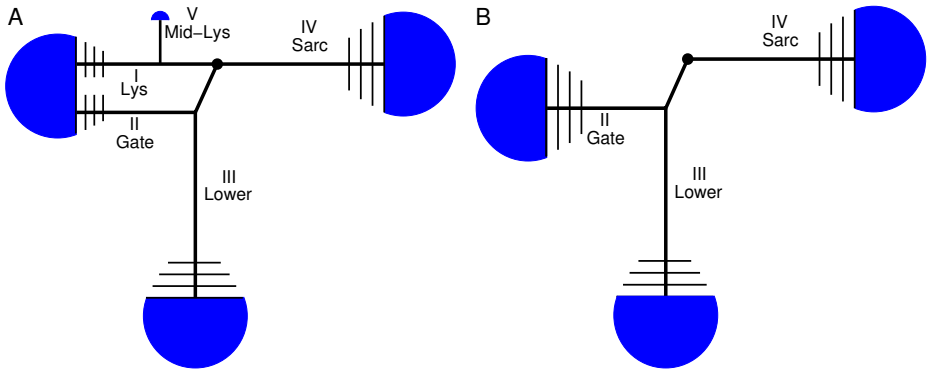

Figure 4.4: Schematic representations of pathways and portals in the (A) apo and (B) bound systems, established by refinement of MFEP's from string method using milestoning MD, as described in the text. The large black dot designates the active site cavity in which the free energy minimum for O_2 is located for both systems. Arcs at each portal signify the spherical portal boundaries.

Analysis of the kinetic network using MMVT is a two-step process. It begins with identifying the first cell on each MFEP which can access the solvent. Once each solvent accessible cell is identified, a solvent spherical shell can be applied to help approximate entry into the bulk. As seen in Figure 4.4 all pathways originate from the active site. Each terminates at the first instance of exit into the solvent. The solvent shell is then applied and further tesselated as shown in Figure 4.5.

Generally, within the protein interior, each cell required 10-50 ns to attain sufficient sampling, as determined by saturation of each $k_{i\to j}$ within a cell. With the interior completed, the solvent portals were examined. Since the free energy gradients are rather shallow over large distances in this region, the MFEP, though calculable, does not really correspond to the center of a compact tube-like transport channel, and we could not do optimal milestoning with Voronoi centers along such an MFEP. Instead, we include TAMD frames uniformly distributed within a sphere of 10 or 12.5 Å

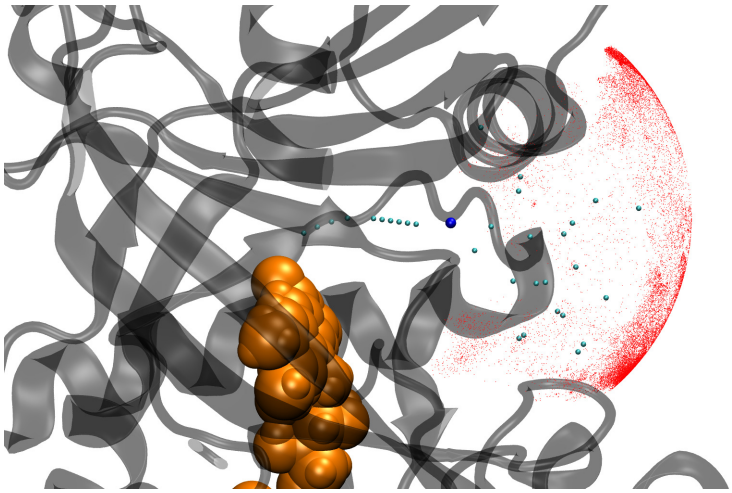

Figure 4.5: Sample portal setup. The Apo-Sarc MFEP is discretized into its interior Voronoi tesselation centers (cyan) with the first cell exiting to the solvent as a larger blue sphere. Cell centers defining the tesselation in a 12.5 Å sphere (cyan) are shown along with the first 5ns of data for any transition out of the sphere (red). Nearby protein structure is shown in cartoon (black) and cofactor in space filling (orange) representations.

centered on the first string image which can exit into the solvent with approximate spacing 2.5 Å. Only frames in the direction of the solvent are kept. Those falling on the MFEP are excluded. These cells are prepared in the same way as the MFEP and run to saturation. The solvent cells have a much larger spacing (2.5 Å) compared to the interior (∼0.7 Å) yet only required between 10-25 ns to saturate.

We began with the apo-OB configuration. The two pathways which exit into the solvent through the cofactor admitting cleft (Gate and Lys) presented a problem. It appeared as though O_2 traveling along either of those pathways could access the solvent even from deep within MSOX and near the sugar backbone of FADH. Large deviations from the MFEP indicate a breakdown in the assumption that our ligand should travel a tube-like path before reaching the solvent at the end of the MFEP. We were unable to confine O_2 to the MFEP without solvent transitions outweighing those along the MFEP, and concluded that with a completely open cleft MSOX is porous and not suitable for analysis with milestoning. Additionally, our computed

entry rates are predicted to be near the diffusion limit for the closed cleft system already, and this is unlikely to be different for a state where O_2 entry/exit is easier by inspection. We therefore removed the open cleft configuration from further analysis.

Conversely, when examining the apo-CB system, it behaved well with only minor pinching that was overcome via combining cells. We observed the Lys and Gate pathways (apo-CB-I and II) in the apo system to have portals that are relatively close together on the protein surface, and overlapped in some areas. We therefore elected to use a single portal sphere to encompass both (Figure 4.4 A), and their contribution to the entry rate is reported as combined. Also in the apo-CB system, during the initial set of milestoning MD simulations we discovered a cell which, though not on the protein surface, could escape into the solvent. This "leaky" cell occurred along the Lys pathway and its corresponding portal was distant from all others, therefore requiring its own portal sphere. We refer to this new portal as "Mid-Lys" (Figure 4.4A) and its contribution to entry and exit is delineated from all others. After MMVT was completed, the apo-CB system consisted of 178 cells including the solvent sphereical shell.

We explored the bound system last. The Lys pathway (bound-I) exits the active site through a channel that was observed to be blocked in portions of the MD simulations by the side-chain of Phe256. This residue is present in two distinct configurations wherein the Lys channel is either blocked or open. We therefore ran adaptive biasing force (ABF) simulations [58] to assess the free energy associated with flipping the side-chain of Phe256 into an open-channel state. As seen in Figure 4.6 the free energy barrier required to open is about 10 kcal/mol. This is a large free energy barrier, one not likely overcome easily under biological conditions. Consequently, we chose to ignore the possibility that Phe256 spontaneously opens and did not perform milestoning along the Lys pathway for the bound system. Examination of the TAMD

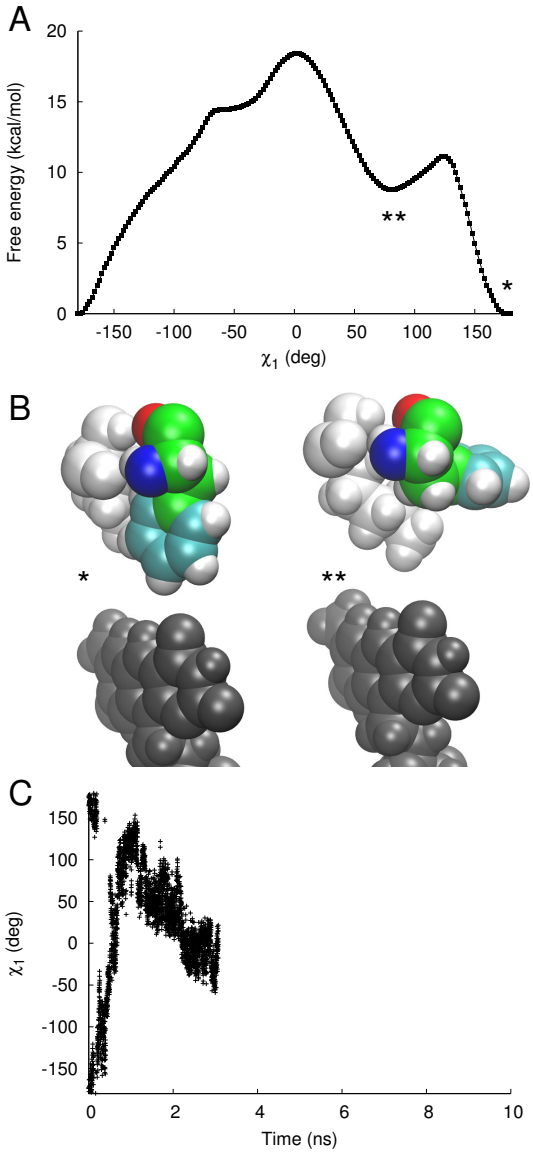

Figure 4.6: (A) Free energy profile from a 10-ns adaptive-biasing force MD simulation sampling the C-C_α-C_β-C_γ torsion of Phe256, χ_1. Open (*) and closed (**) configurations are indicated. (B) Snapshots from the ABF trajectory illustrating the closed (left) and open (right) configurations of the Phe256 sidechain. Only atoms in Phe256 (colored based on atom name), the flavin (grey), and Lys265 (white) are shown. (C) Trace of Phe256 χ_1 vs. time in the ABF calculation, illustrating that χ_1 has repeatedly sampled its domain.

trajectories indicated that flipping open occurred primarily when O_2 was pinned between the inhibitor and Phe256, causing it to reorient and allow O_2 to pass. Thus, analysis of the bound system was limited to three of the four initial MFEPs. After

MMVT was completed, the bound system consisted of 177 cells including the solvent sphereical shell.

In summary, the main points for kinetic network refinement are:

1. Five pathways in apo-CB encompassing 178 cells total.
2. Three pathways in bound encompassing 177 cells total.
3. Removal of the apo-OB system from consideration since it is not well suited to MMVT analysis.
4. Removal of the bound Lys pathway since flipping open of Phe256 is not likely to occur under normal conditions.
5. Combination of the Gate and Lys solvent spheres in apo-CB MSOX.
6. Addition of the Mid-Lys pathway in apo-CB MSOX.

4.2.5 Raw Milestoning Data

To give an impression of the amount of simulation data that is processed, and to provide a view of the 3D structure of the pathways, we show milestoning MD hitting points on all milestones along the pathways in Figure 4.7 for the bound system. We show only the first segment (5 ns) of each cell on the interior of inhibitor bound MSOX for clarity. No solvent data is shown to emphasize the interior hitting which is generally centered around the MFEP. The milestone planes are also easier to differentiate. Note that each plane has hitting points that arise from collisions occuring in two MD simulations, those in cells on either side of the milestone. Transition attempts can occur off the MFEP, as seen in the hitting on the Sarc pathway and the end of the Lower. Generally, the closer O_2 came to the solvent, the more dispersed hitting became, as the solvent influenced motion more significantly. A total of 355

cells were run between the apo and bound systems, from anywhere between 10−50 ns each. Each segment of 5 ns required approximately 8 real hours on two nodes (16 processors each) on the supercomputer Stampede at the Texas Advanced Computing Center (part of the XSEDE consortium [53]). Averaging the total time for each cell we arrive at approximately 15 ns per cell. This amounts to approximately 270,000 CPU hours. MMVT could not have been performed without the aid of the TACC supercomputers, as local simulations with 16 processors and GPU acceleration required about 16 hours per segment in each cell. For comparison, this amounts to roughly two cells per week provided no pinching or instability arises. The total time to run all cells would be nearly 3.5 years (in an ideal case) versus the 4 months (including troubleshooting instability) it took to run on the TACC supercomputer Stampede. This is mostly because local work could only accomodate one job at a time, while the TACC supercomputers handled up to 50.

4.2.6 Rates and Mechanisms of O_2 Entry

In Figure 4.8, we summarize the entire set of milestoning results in the form of two kinetic networks, one for each of the apo and bound systems, that represent individual pathways of O_2 entry into and exit from the MSOX hydrophobic cavity on the *re*-face of the flavin ring. For these individual representations, the Markov state model defines the active-site macrostate as the node all pathways extend from. Each portal macrostate is the set of solvent milestones that interface the solvent at each respective portal. Each arrow is labeled by the MFPT in μs. We first consider the entry pathways.

In the apo system (Figure 4.8 left), the dominant pathways carrying O_2 from the solvent to the active site are Gate/Lys, Mid-Lys, and Sarc. The MFPT's from solvent portal to the active site are about 1190, 1886, and 842 μs for the Sarc, Gate/Lys, and

Figure 4.7: Representative selection of milestone hitting points (red) for the bound system, with the protein represented in black. Major pathways (cyan) and their portals are also labeled. The first 5 ns of data is shown in each cell.

Mid-Lys pathways, respectively, at an O_2 concentration of 209 μM, corresponding to saturation at 37°C and 1 bar in equilibrium with air. Hence, in the apo system, we see that the protein is essentially porous, with three entry pathways that are all significant contributors to the overall rate of entry. It is interesting to see that despite being a poor system for MMVT analysis, the conclusion is the same for both apo-OB

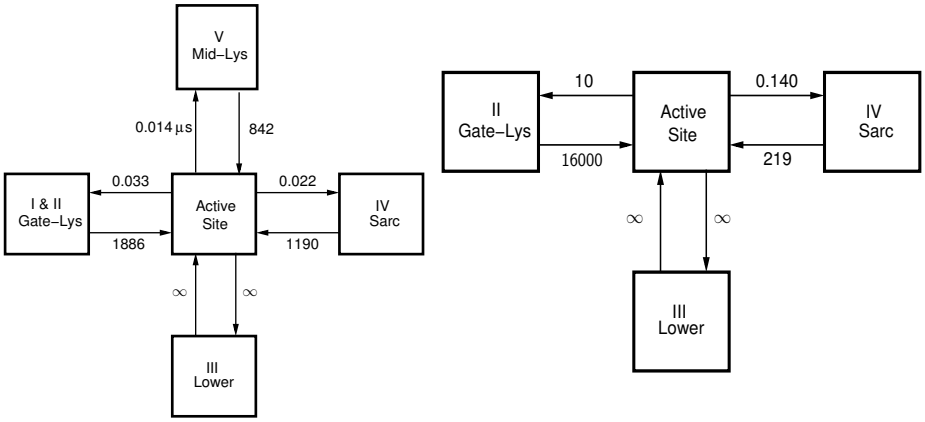

Figure 4.8: Kinetic networks for the apo (left) and bound systems (right). Labels indicate MFPT in μs. Entry times correspond to entry from aqueous solution at an O_2 concentration of 209 μM.

Table 4.1: Overall entry and exit MFPTs for apo and bound MSOX at oxygen saturation concentration (209 μM) in water in equilibrium with air at 37°C.

	Apo	Holo
Entry	1553 μs	590 μs
Exit	10 ns	138 ns

and apo-CB that MSOX is porous.

In the bound system (Figure 4.8 right), the situation changes markedly. Here, the MFPT along the Sarc pathway is greatly reduced while that along the Gate pathway increases so much it does not contribute significantly. Also, as noted earlier, the bound system does not have a Lys or Mid-Lys pathway thanks to Phe256. Therefore, in the bound system, O_2 predominantly enters by the Sarc pathway. In both systems, the MFPT along the Lower pathway is so large that it is effectively infinite, meaning that although there is a path there, it does not contribute at all to the O_2 entry rate in either system.

The overall second order rate constant for entry can be computed at any O_2 concentration, and it is observed to be essentially invariant across O_2 concentrations

(as expected) for both the apo and bound systems. Calculation of second order rate constants is based on the overall MFPTs presented in (Table 4.1) since these MFPTs encompass the entire interconnected network, rather than a lone pathway. Interestingly, although the pathways shift in the weight of their contributions to the overall entry rate between the apo and bound systems, the overall rate itself is not strongly affected by substrate-mimic binding. We predict that $k_{\text{entry}} = 8.1 \times 10^6$ M^{-1} s^{-1} and $k_{\text{entry}} = 3.12 \times 10^6$ M^{-1} s^{-1} for bound and apo, respectively. This value is about an order of magnitude higher than the second-order rate constant for oxidation of reduced wild-type MSOX, $2.83 \pm 0.07 \times 10^5$ $\text{M}^{-1}\text{s}^{-1}$, determined during mutagenesis studies [9] (i.e., k_3 in Figure 4.1) but about the same as the second-order rate constant for O_2 consumption by the most active flavoenzymes [5].

Since k_{entry} for both apo and bound MSOX is not significantly different, and because $k_{\text{entry}} > k_3$ in both cases considered here, it is unlikely that entry of O_2 limits the rate of flavin oxidation. It should be borne in mind that we define entry as an event in which an O_2 transits from the solvent to the local free-energy minimum associated with the active site, as computed using single-sweep reconstruction, and that the standard molecular mechanics force field used (CHARMM) cannot model specific charge-transfer interactions. The oxidation steps in the standard mechanistic picture can therefore be subdivided into more fundamental steps of entry to the active-site cavity, acquisition of a specific O_2/flavin complex (likely involving polarization), then electron transfer to complete the reaction. Our second-order rate constant corresponds to the first of these fundamental steps, and given that the overall rate of the oxidation step is slower than our computed entry rate, we conclude that the fundamental step of entry is not rate-limiting.

4.2.7 Rates and Mechanisms of O_2 Exit

We look now at the individual exit rates for apo and bound MSOX. Oxygen exit is significantly faster in the apo state since exit times are on the order of nanoseconds rather than microseconds. We also note that no one path clearly contains the majority of the flux. As with entry, the Sarc, Gate/Lys, and Mid-Lys pathways all exhibit comparable exit MFPTs at 22, 33, and 14 ns respectively. Once again we see the apo state exhibiting porous behavior.

In the bound system, we see one dominant exit route. The Sarc pathway displays a MFPT of 138 ns, nearly two orders of magnitude faster than Gate at 10 μs. As with entry, exit MFPTs on the Lower pathway are so large as to be considered infinite, effectively not contributing to the overall exit of O_2 in either state.

Based on the overall exit MFPTs we find $k_{\text{exit}} = 10^7 \ \text{s}^{-1}$ and $k_{\text{exit}} = 7.2 \times 10^6 \ \text{s}^{-1}$ for apo and bound respectively. Differing by an order of magnitude, it is more likely that exit is limiting in the case of O_2 oxidation of the flavin. A longer residence time within MSOX increases the likelihood of collisions with the flavin isoalloxazine ring, required for reduction of molecular O_2 and subsequent reoxidation of the flavin. This is further supported by the fact bound MSOX shuts down all but one exit route, the so-called Sarc pathway. Evolution appears to have led MSOX to be porous without a substrate in the active site, while ensuring the highest probability of O_2 being in contact with the isoalloxazine ring when a substrate is present in the active site.

4.2.8 Structural Basis for Change in MFPTs

In comparing the bound and apo states, there are no major structural differences within the vicinity of the flavin isoalloxazine ring, save the presence of the inhibitor FOA itself, that lend themselves to an easy explanation for the shutdown of the Gate pathway in the bound state. Our long MD trajectory does not show any opening

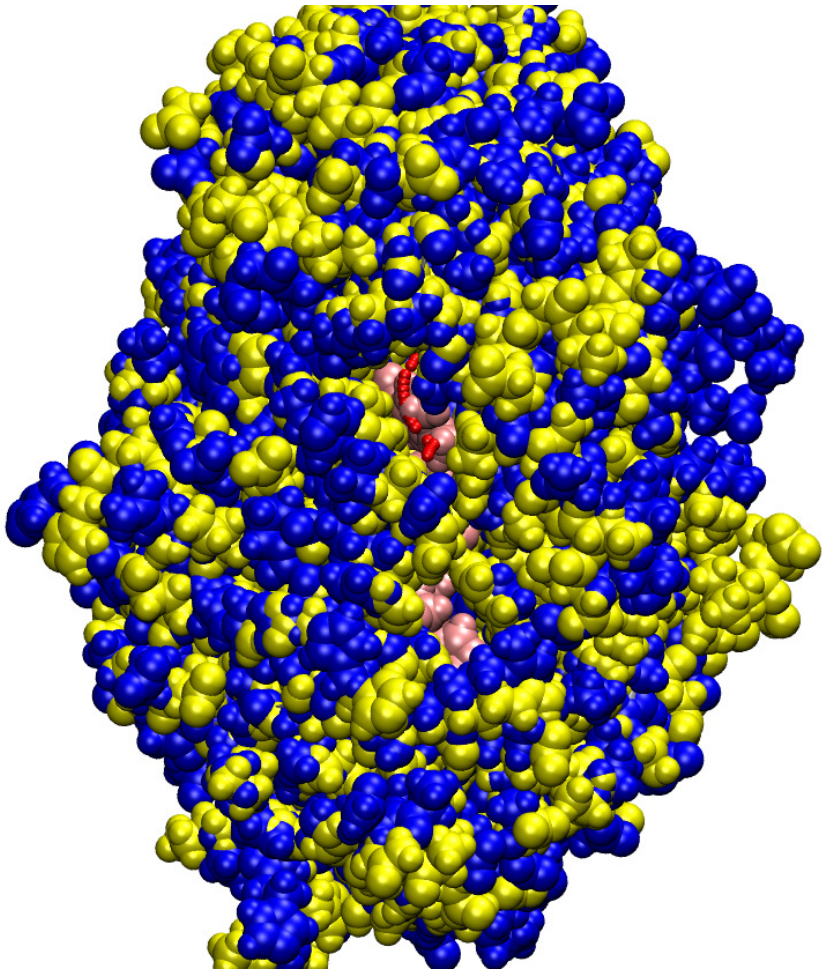

Figure 4.9: Aligned holo (yellow, space filling) and apo (blue, space filling) structures showing the relative states of the cofactor admitting cleft during equilibrium MD. The cofactor (pink, space filling) is shown as well as the apo-Lys MFEP (red, spheres). The holo structure is tightly packed around the cofactor, while apo is more relaxed.

or closing of residues which would facilitate or hinder the progress of O_2. However, further out from the ring, we see the cofactor admitting cleft exhibits two unique configurations when comparing bound and apo MSOX. It remains tightly closed in the bound state, yet is less tightly packed in the apo state, as shown in Fig. 4.9. There is no pathway analogous to the apo state's Gate path in the bound state that does not overlap residues. This likely explains why the only MFEP identified through this portal in the bound state does not permit transit during milestoning MD, as described

previously in regards to Phe256. We therefore see a direct effect of inhibitor binding on multiple transport pathways.

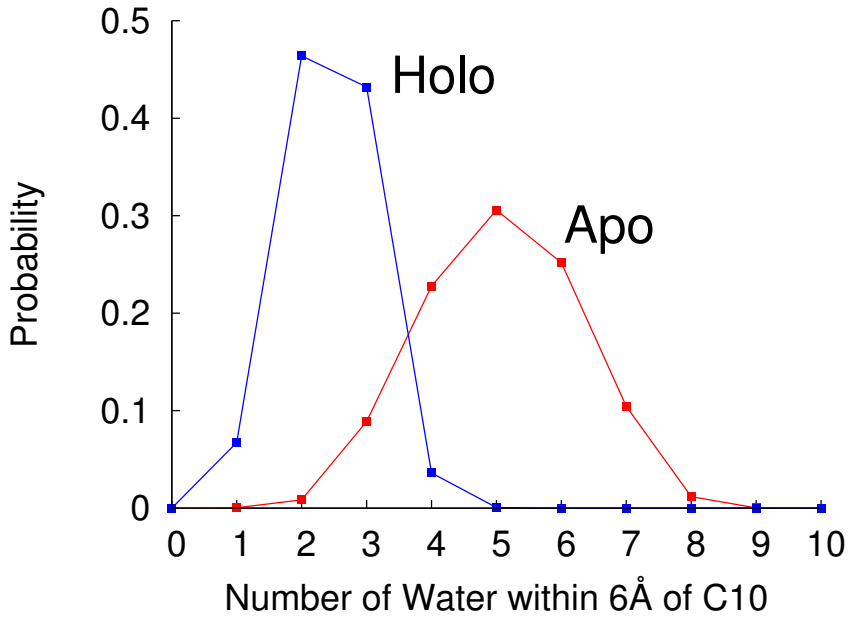

Figure 4.10: Water occupancy distributions in the active site (defined as sphere of radius 6 Å centered at atom C_{10} of the flavin ring) for apo and bound MSOX.

Examination of the portal-specific MFPT's indicates inhibitor binding effectively closes off all pathways available in the apo state except one: the Sarc pathway. Yet along this pathway, the rate of entry almost doubles relative to the apo state. This may be explained by the fact that the presence of the inhibitor desolvates the active site, excluding approximately three molecules of water that easily access the *re*-face. Addition of O_2 to the *re*-face removes another water molecule, resulting in a net loss of three to four water molecules from the active site, as we show in Figure 4.10. Desolvation at the active site has also been proposed as important to reoxidation of the flavin in other flavoenzymes [5]. We observe that the more hydrophobic environment

in the binding site created by the inhibitor speeds the entry of O_2 along the Sarc pathway, supporting this notion.

4.2.9 Modified Ping Pong or Ping Pong?

We have used Markovian Milestoning MD simulations to compute entry and exit kinetics of O_2 in MSOX. The overall second-order entry rate constant slows by a factor of about 2.5 upon inhibitor binding, while exit rates slow by nearly an order of magnitude. The net effect is the prediction of increased O_2 residence time in the active site when the inhibitor is bound relative to the apo case, which provides indirect support for the modified ping-pong mechanism. We observe that binding of the inhibitor shuts down multiple routes of entry and exit, yet speeds up entry along the dominant channel. Experimentally, linear regression analysis at different O_2 concentrations with the native substrate sarcosine present show a systematic variation in slope with O_2 concentration. This is consistent with a modified ping pong mechanism wherein O_2 reacts with the reduced enzyme prior to product release [1]. Combined with the substrate-mediated desolvation we observe, these two pieces of evidence suggest that is is kinetically favorable for flavin reoxidation in MSOX to operate via the modified ping-pong mechanism.

5. Preliminary Investigation of Whether Voronoi Cells Can be Used to Generate Correct Pathways

5.1 Introduction

In single sweep, we sought to uncover the interconnected network of O_2 transport pathways within MSOX. This required a significant computational expense in the form of force calculations wherein 1,033 individual simulations were performed. Not only was the computational expense large, but the physical time as well. Similarly, application of MMVT to the MFEPs revealed by single sweep required 355 individual cell simulations, requiring up to 10 segments each. While computing time was similar to the force calculations, the physical time required for this step was several times that of the force calculations. The primary reason for the delay was the constantly changing tesselation that resulted from pinching instability.

While performing MMVT, we noticed that our assumption on MFEPs carrying the majority of the flux was not always accurate. Indeed, some regions displayed markedly different transport pathways than anticipated from the MFEPs. In fact, on the Sarc pathway in the bound system, we needed to remove an entire curved segment of the tesselation and replace it with a straight line to achieve stability. After observing this behavior, it was clear that cell transitions can be markedly different from the MFEP. We began with approximately 230 interior cells for each of the three systems in question. Removing apo-OB entirely since it was ill suited for MMVT, we were left with a starting point of 460 cells total. In the end, the remaining cells were pared down to 355 total, *including* the solvent sphereical shells. While we found iterating constantly until the tesselation presented a stable network to be the best course of action, it was very time consuming. Thus, we began to question whether it

was possible to do the analysis in a more efficient way.

We observed the problematic cells along the Sarc pathway to attempt to converge to the most stable configuration. However, since the tesselation involved highly curved portions, crashing resulted from pinching. In this section, we kept removing and merging cells until we eventually settled on a straight line rather than a U-shaped one in that region. Clearly, it was possible that the dynamics could guide the shape of the entry/exit channel without prior knowledge.

Seeing this we began to wonder if it was possible to achieve a stable tesselation from the onset of MMVT. Specifically, could we begin with some arbitrary starting tesselation, and allow it to update itself. The idea draws inspiration from on the fly string method [59] wherein multiple system replicas are run at the same time and allowed to update in real time while still subject to the reparameterization step. Traditional string method [35] in contrast requires multiple replicas to be run until the net forces acting on each image saturate, at which point a position update occurs. Unfortunately, both traditional and on the fly string require significant computational effort due to their iterative nature, and thus immense physical time as well. Since computational effort scales with system size, convergence is often slow.

Our hope is to combine on the fly string method with MMVT to develop a new technique with an evolving tesselation that can identify relevant pathways. These pathways would not be MFEPs, but rather principle curves describing the pathways of greatest flux in a given system. After convergence of the tesselation, we would have a stable system ready for MMVT which would be free of extreme ballistic pinching. Without extreme ballistic pinching, extraction of MFPTs would be much faster than the current iterative procedure. We also aim for it to be more efficient than either string or MMVT, as updates will ideally be done based on minimal statistics, leading to very fast position updates. In this way we can remove the overhead associated

with single sweep, as well as the iterative network refinement in MMVT and string method.

5.2 Methods

In order to test whether we could successfully identify these paths of greatest flux, a simple system was needed. Previously, our group performed Transition Path Theory (TPT) calculations on a three-well potential using finite elements [60]. The potential has the form:

$$V(x,y) = 3e^{-x^2-(y-\frac{1}{3})^2} - 3e^{-x^2-(y-\frac{5}{3})^2} - 5e^{-(x-1)^2-y^2} - 5e^{-(x+1)^2-y^2} + 0.2x^4 + 0.2(y-1/3)^4 \tag{5.1}$$

This potential is well suited for testing for several reasons. First, it is lower dimension than an all atom system. We are dealing with one particle in a box rather than thousands of interacting atoms. Local work is therefore possible and extremely fast. Additionally, depending on the system temperature, the potential exhibits two distinct dominant transport pathways between wells. At low temperature ($\beta = 6.67$), a particle cannot overcome the small barrier separating the two deeper wells at (0,0). Therefore, a successful transition will move from one deep minimum and travel through the shallow minimum at (0,1.5) en route to the other. If the temperature is raised high enough ($\beta = 1.67$), the particle has enough energy to overcome the small barrier and can transition directly between the two wells without utilizing the shallow one. This phenomenon is referred to as entropic switching. Lastly, this system has been studied before using TPT [60, 61] and we have exact MFPTs to compare against for gauging accuracy and efficiency of the new method.

We began with a simple 2D FORTRAN molecular dynamics code developed by our group. I then added the three-well potential, and further modified the MD code

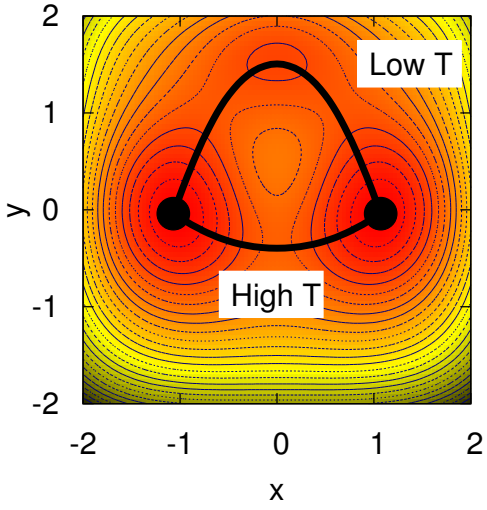

Figure 5.1: Three well potential. Depending on the system temperature, one of two paths should dominate transport between the two large wells at (1,0) and (-1,0). Contour lines show regions of constant value, and colors probability, with red as regions of high probability and yellow/black as much lower probability.

to support Voronoi cells by implementing position and velocity rewind/reversal based on a distance check. We now have a simple molecular dynamics system consisting of one particle moving around in 2D on the three-well potential. The system would check for Voronoi cell transitions with the addition of a flag to the input file. In this way, should the need for the three-well potential arise, it can be used for other tests.

Armed with a suitable test system, we now needed to develop a viable experiment. To be successful, we need to properly identify all relevant pathways, and do it faster than the combination of single sweep and MMVT. We chose to assess pathway determination first. Similar to string method, we need a beginning and end point on which to build the initial tesselation. We chose the two deep wells located at (-1,0) and (1,0), and manually created strings with 5 and 11 cells equidistant along the x-axis. The reason for two different sets of images is that with such a small system,

there is no way to know beforehand what an appropriate cell spacing should be. We also chose to begin with a low temperature ($\beta = 6.67$) as it is the simpler case. Only one pathway should present itself, the upper arc (Figure 5.1 Low T) traversing all three wells.

Each Voronoi cell was initialized the same way as the MSOX system, with the particle placed exactly in the center. We then ran each cell for 10,000,000 time steps so that with sufficient data we can identify the minimal time required for a position update. Similar to MSOX, output consisted of the individual x,y points where a transition occurred. Since simulation time was relatively fast, we also performed several tests where the location of the particle at every step was logged. Using this data , we then sought a way to update the Voronoi cell center position, and iterate until converging on a non-fluctuating pathway.

5.3 Preliminary Results

Since the starting configurations were determined manually, simple cases were studied first. Recall that our goal is to begin without prior knowledge of the potential energy surface. Also, without prior knowledge of the potential, the most obvious way to connect the wells is with a straight line. We established two test systems with 5 and 11 cells total, corresponding to cell centers every 0.5 and 0.2 along the X-axis, respectively. Each starting configuration began at (-1,0) and ended at (1,0). These points represent the two deep wells in the potential and are the "reactant" and "product" states. Since reactant and product states are invariate in string method and MMVT, no MD was run there. After running 2D MD in each of the interior cells, we needed to find a way to analyze the results to determine a position update. Based on the position update, we could then assess whether the predicted pathway was consistent with previous work.

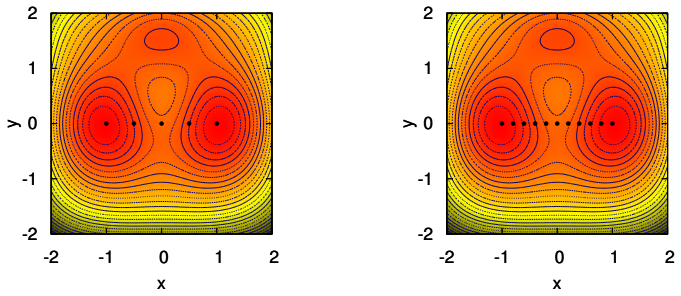

Figure 5.2: Starting cell centers for the 5 cell (left) and 11 cell (right) systems.

This task of determining an appropriate position update turned out to be more complicated and technically challenging than expected. We found that many ways exist to update the tesselation. The following choices were initially considered:

1. Update based on all boundary collisions.

2. Update based on transition attempts, i.e. hitting wall i and then $j \neq i$ as the next boundary collision.

We began with the simpler five cell system. Extracting the data for both all boundary collisions and only transition attempts, we noticed very sparse data in the transition attempts, particularly in the region of the barrier at (0,0). This is likely the result of our choosing the lower temperature as the starting point. Our tentative updating scheme then became determining where the average of all hitting points was on either side of a cell boundary, and then using those two points, calculate the midpoint as the new cell center. Then, approximating the new string as a series of line segments, reparameterize so that each cell center distance was equal. Using this approach however, we quickly realized that as the string evolved, pinching would cause a buildup of hitting points, resulting in evolution the opposite way, leading to a constantly fluctuating string that never converged. We also attempted to use

the position of the particle at every time step and determine its average position to update the tesselation. This approach had similar non-convergence. Since we had very little transition data to try alternate methods on, we ended our analysis by concluding there were to few cells to gather data from.

During our examination of the 5 cell system, we also determined that using all the boundary collisions as a basis for updating was redundant. Ultimately, we aimed to model a reactive trajectory going from one deep well to the other. Use of all boundary collisions already assumes a key tenet of such a reactive trajectory, namely that transitions occur leading from one well to the other. Therefore, using the transitions alone should provide the same results. Despite this observation, we still examine both all hitting and all transitions on the 11 cell system for consistency, understanding that they should yield similar results.

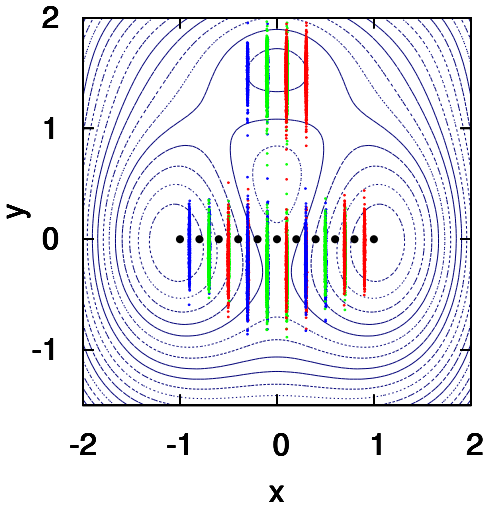

Figure 5.3: All transition attempts for the 11 cell system at low temperature. The potential energy surface is removed for clarity, but contour lines remain. Hitting is shown in alternating blue, green, and red to show different cells.

Figure 5.3 illustrates the data we used for testing from the initial configuration of the 11 cell system. Clearly, both all hitting and all transition data should indicate that the same regions are sampled, but the transition data further narrows it down to two distinct regions. These results are unexpected since we expect only one pathway to present itself in the low temperature case. After much discussion we hypothesized that similar to the MSOX system, all routes are always present, but not all contribute the same. There is a definite probability for the lower pathway to exist, it is just very low (in fact it should be very close to zero) compared to the upper. Thus, the analysis now centered around whether this new method could identify the correct dominant pathway and whether or not identification of the secondary pathway was accurate.

For consistency, we peformed a position update identical to the 5 cell system, wherein we identified the average hitting or transition point on either boundary. We then calculated a midpoint and updated the cell center. Finally, a new tesselation was generated from reparameterization of the series of line segments. While this approach was fine for most cells, those which clearly sampled two unique regions displayed a cell update that did not make sense. It is incredibly unlikely that the correct position update is the middle of the two sets of points, where very few transitions or hits occur. It also reveals nothing about which path is dominant or that two seem to exist. We concluded that simply averaging the hitting or transition attempts did not provide enough information to proceed.

We then turned to analysis of the transition data using histograms. From the transition attempt data, we generated a histogram along every cell boundary. For the initial configuration, this is quite simple as all boundaries are vertical lines. As expected, the regions where two sets of hitting points occur display two peaks, as seen in Figure 5.4. Cells 5, 6 and 7 exhibited this behavior. Clearly, with multiple distinct large peaks, there must be more than one pathway. The largest peaks also

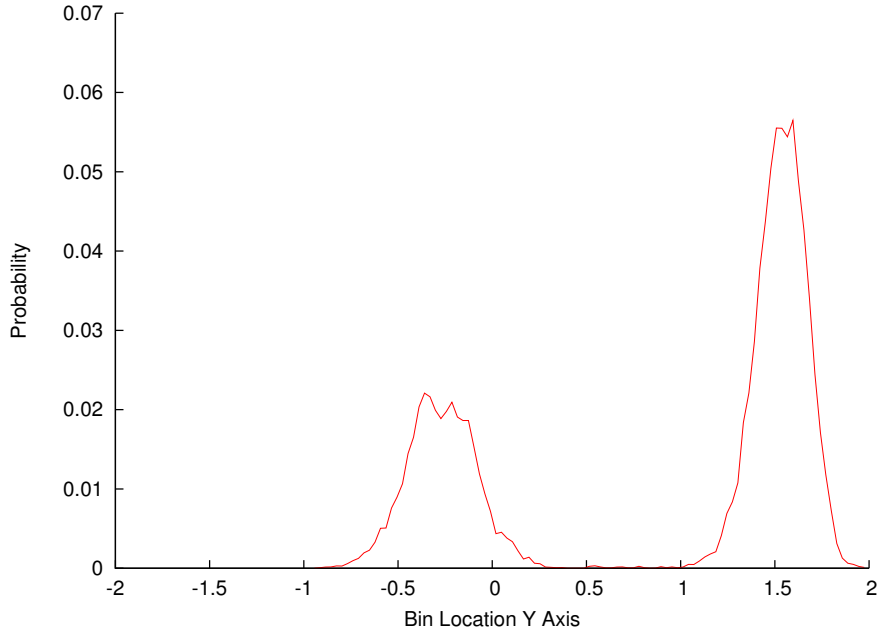

Figure 5.4: Histogram for data corresponding to transitions from cell 6 to cell 5. Two distinct peaks are observed, indicating two distinct pathways.

correspond to where we expect to find the upper and lower pathways described from previous investigations. However, we cannot determine precisely which is major or minor, but attempt to infer based on the relative heights of the largest peaks which route is more likely. We observe the peak corresponding to the upper pathway to be approximately three times larger than the lower, thus concluding the upper would be preferable at these conditions. We must still consider the lower pathway since it is evident in our data, although all indications from the literature are that it should be absent [61]. For this reason we are not certain whether using just a histogram is the best way to examine this system. Despite this, it seems possible to identify the two routes between wells even at a low temperature.

Recall that our goal is to update the tesselation in real time, so that ideally the simulation ends when convergence is reached. Analysis of the histogram required a user choice wherein we physically looked at the distribution and found where poten-

tial paths should exist. Can we potentially automate this process to achieve such results? One such way to automatically update is by identifying the largest peak in the histogram. While this is trivial to automate, it does not allow us to identify multiple pathways. Instead, we used clustering. Here, we would first generate the histogram as before. Then, a program would identify the smallest nonzero bin and set it as an increment. The program would then remove all bins below integer multiples of that increment, until the largest set of bins was left. The midpoint of this set of bins would be the dominant peak. In this way we can also set whether to include other "large" bins based on relative heights or distances between large peaks. As implemented currently, we only identify the largest set of bins for updating position.

A small caveat to our approach is that after the initial iteration, we will not have straight lines representing the boundaries anymore. Now, we will have sloped lines in 2D space (or planes in 3D) which are not simple to realize with a histogram. We therefore needed to implement a coordinate transformation on the data to ensure it was properly analyzed.

A custom group developed 2D MD code modified by me, implementing a Voronoi tesselation, was used to generate the hitting data. TcL codes were built for data extraction from the log files from the MD code. C and FORTRAN codes were developed to average the hitting points, determine midpoints, and generate new positions through reparameterization of the line segments. FORTRAN codes were written to generate histograms, perform clustering, change coordinates, and check whether our data followed the correct Boltzmann distribution for this potential. Multiple bash scripts were written to interface the various codes. See Appendix D for a listing and description of these codes/scripts.

We are still investigating whether the prescence of a second large peak is relevant or an artifact of the methodology employed thusfar. In this way, results from our

investigation of this test potential have been inconclusive so far, but we hope to launch it as a full research project in the near future. Removing much of the overhead from single sweep and milestoning could vault this technique into the forefront of determining entry and exit rates as well as entry and exit pathways for a multitude of systems.

6. Conclusion and future work

6.1 Thesis summary

In Chapter 3, Aim I, we used the composite method of Maragliano et al. [29] to map pathways and sites relevant for O_2 transport in MSOX as well as to gain a better understanding of the impact of substrate binding on these pathways. We observe two O_2 binding sites near the reduced flavin ring on the *re*-face, suggesting that O_2 reduction is at least initiated on the same side of the flavin ring on which sarcosine oxidation occurs. Generally, we observed that multiple plausible pathways exist by which O_2 can access the sites in the *re*-face cavity, regardless of whether or not the substrate-mimicking inhibitor 2-furoic acid is bound. They are geometrically similar in the three different systems examined (apo, bound, and apo with an open flavin cleft), and present many localization sites for oxygen in the same locations. We do note that generally the bound system contained more shallow minima, and that those in similar locations to the apo states were lower in free energy. This leads to the conclusion that bound MSOX should transport O_2 slower, but further investigation using milestoning was necessary to test this hypothesis. We have thus shown that the composite method of Maragliano et al. [29] can be easily adapted to study oxygen diffusion pathways in the flavoenzyme MSOX.

In Chapter 4, Aim II, we explored these pathways using Markovian Milestoning in Voronoi Tesselations (MMVT). Our aim was to refine the network determined via the composite method, and compute entry and exit rates. We then compared against experimentally determined values corresponding to the step which could identify whether MPP or PP was the dominant mechanism. Though the MFEPs were geometrically similar, marked differences in the free energies between ligand-bound

and unbound states as determined by the composite method were observed. Further structural analysis showed that fluctuations in relative loop placements and side-chain orientations among the three systems explained the appearance of these free energy minima. However, in order to fully characterize the kinetic network, we applied MMVT to pare down the system to its' most important pathways and segments, and applied a solvent sphereical shell to approximate entry into the bulk. The computed overall second-order entry rate constant slows by a factor of about 2.5 upon inhibitor binding, while exit rates slow by approximately an order of magnitude. The net effect is the prediction of increased O_2 residence time in the active site when the inhibitor is bound relative to the apo case, which provides indirect support for the modified ping-pong mechanism. We also discovered that binding of the inhibitor shuts down multiple routes of entry and exit, essentially funneling O_2 through one dominant pathway. Combined with substrate-mediated desolvation on the *re*-face, this evidence suggests that it is kinetically favorable that flavin reoxidation in MSOX operates via the modified ping-pong mechanism.

In Chapter 5, we investigated the potential for removing the overhead associated with mean force calculations and the iterative network refinement of MMVT. We attempted to develop a method inspired by MMVT and on the fly string method [34, 59] which can update a blind Voronoi cell system in real time to converge upon pathways of greatest flux. Our results indicate that simple averaging of the boundary collision events is insufficient to permit string updating. Use of histograms to determine regions of preferable hitting showed promise, but identify multiple pathways in a case where only one should exist. Despite this, we successfully developed a way to automate updating of the dominant peak in the histogram.

6.2 Future computational work

Our work has shown the composite method of Maragliano et al. [29] can be adapted to the MSOX system. By extension, the implication is it can be readily applied to any number of systems, even those without complete parameter sets already built. The addition of Markovian Milestoning in Voronoi Tesselations completes the story begun by the composite method, wherein we compared directly against experiment. It has also been used to study myoglobin by our group [2], characterizing multiple pathways for CO diffusion as well as the ensuing kinetic network. We now hope to apply the combination of MMVT and the composite method to study many more biological systems, including other flavoenzyme oxidases (GOX, TSOX, HSOX, etc) as well as larger systems such as hemoglobin. With the addition of rates which can be compared against experimental values, it is possible to model difficult to observe phenomenon such as cooperativity in hemoglobin. Additionally, we can study other flavoenzymes to assess whether similar mechanisms modulate reactant transport and enhance our knowledge of flavoenzymes in general.

There is still some work to be done with MSOX however, as our work focused on comparing inhibitor bound and unbound states. As with any good model, we can now relax some of the assumptions employed here, to ultimately arrive at more accurate values. For MSOX, this can begin with building a model containing the native substrate sarcosine (N-methylglycine) and developing suitable parameters for it. The entire body of work can be repeated then, to assess whether the loss or gain of specific pathways is linked to a specific species present in the binding site. We can also change the force field so it is polarizable or reactive, allowing us to model the superoxide, and perhaps one day even the reaction itself. In this way we can also observe whether O_2 activation is preferable on the *si*-face, and what steric effects really occur in the binding site due to the presence of multiple components.

6.3 Potential Experimental Validation

In addition to the catalytically relevant residues (Lys265, Arg49, Cys315), our results from single sweep and milestoning implicate residues and regions of MSOX which have not been explored experimentally. Near the active site, it was suggested that Phe256 would be important to stablization of the superoxide as well as helping to define a pre-organized binding site on the *re*-face [14]. Our results show that Phe256 is an integral part of a larger hydrophobic pocket exhibiting two O_2 minima on the *re*-face, as well as taking two distinct configurations which can either permit or block O_2 entry and exit to the active site. Experimental confirmation of the importance of Phe256 would enhance our knowledge of flavoenzymes by illustrating the importance of a large nonpolar residue in close proximity to the catalytically active residues. This could be done via mutagenesis studies wherein Phe256 is mutated to a much smaller nonpolar residue such as Val or charged residues like Lys.

Moving away from the binding pocket itself, we discovered that Tyr55 and Glu57 appeared to influence transport along the large substrate entryway. As this pathway is the only one that is evident in the crystal structure, and precise knowledge of how it controls access of the different substrates and products is key. Through similar mutagenesis studies where smaller residues are substituted, we can hope to see changes in the overall rate of reaction, indicating that these two residues are crucial to transport. Suggestions include using Val or Ile for Tyr55 and Gly for Glu57. With the marked difference in free energy along the MFEP that we observed, it is likely that they have a more drastic effect on the native substrate sarcosine, something previously unexplored by experiment.

Lastly, we observed opening and closing of the cofactor admitting cleft, designated by the state of the Asn41 and Arg282 bridge to affect transport. An open conformation allowed O_2 to diffusively move through MSOX, while a closed conformation

caused O_2 to travel measurable pathways to the active site. One can wonder why the closed state even exists, since the open state is key to both access of the cofactor as well as O_2 to the interior. Mutating one or both of these residues to force the cleft open or closed can show researchers why the cleft needs two states. Likely, it is that coordinated entry of the cofactor with closing of the cleft allows the Cys315 linkage to form, since it could otherwise diffuse out before the bond forms. Additionally, while open, O_2 may not sponge from solution as the inhibitor bound state suggests. These are just two of the questions which can be answered from examination of the cofactor admitting cleft.

6.4 Thesis impact

The simulations in this thesis emphasized the importance of sampling in uncovering transport pathways for O_2 (Aim I) and then in building a Markov state model to define entry and exit rates (Aim II). It also asserts the need for continuing development of existing methods to reduce simulation time and develop stable, robust models. Specifically, this thesis asserts the following:

1. From extensive interior sampling with TAMD, we can determine the free energy of O_2 as a function of position within MSOX with single sweep. This free energy is sensitive to binding of a competitive inhibitor, most notably within the electrostatic cutoff for interactions with the inhibitor. The bound state also shows many more metastable locations. Examination of these metastable locations and their interconnectedness with string method reveals at least four routes exist that transport O_2 to the active site in MSOX. We assert that two local minima exist on the *re*-face, and none on the *si*-face suggesting that reaction and oxygen activation may occur on the same side of the flavin isoalloxazine ring.

2. The four routes connecting O_2 with the solvent and active site can be further analyzed using MMVT to develop a Markov state model. Determination of MFPTs reveals several as unimportant or highly unlikely to transmit O_2 to the active site. The substrate entryway (bound and apo-CB) and the cofactor admitting cleft (apo-CB) dominate transport to the active site. Further manipulation of the MFPTs allows estimation of second order rate constants describing entry and first order rate constants describing exit. We assert that the order of magnitude difference in exit implicates exit as rate limiting for O_2 transport in the two systems. This, coupled with the presence of multiple deep local minima and a hydrophobic pocket on the *re*-face lend support to the MPP mechanism.

3. We assert that the force calculation and iterative network refinement steps constitute a signficiant time barrier in determining rates. Primarily, time was lost through simulation instability that required an iterative approach involving much repeated data to overcome. As such, this limits the usefulness of our approach. We attempt to remove the overhead associated with these steps through a method inspired by on the fly string and MMVT. It is possible to identify a dominant trajectory through histogram based analysis of the boundary collisions. However, the presence of unexpected secondary pathways warrants further testing and refinement of this methodology.

Bibliography

[1] Wagner, M. A.; Jorns, M. S. *Biochemistry* **2000**, *39*, 8825–8829.

[2] Yu, T.-Q.; Lapelosa, M.; Vanden-Eijnden, E.; Abrams, C. F. *J. Am. Chem. Soc.* **2015**, *137*, 3041–3050.

[3] Massey, V. *J. Biol. Chem.* **1994**, *269*, 22459–22462.

[4] Fitzpatrick, P. F. *Arch. Biochem. Biophys.* **2010**, *493*, 13–25.

[5] Mattevi, A. *TRENDS Biochem. Sci.* **2006**, *31*, 276–283.

[6] Trickey, P.; Wagner, M. A.; Jorns, M. S.; Mathews, F. S. *Structure* **1999**, *7*, 331–345.

[7] Wagner, M. A.; Trickey, P.; Chen, Z.-W.; Mathews, F. S.; Jorns, M. S. *Biochemistry* **2000**, *39*, 8813–8824.

[8] Zhao, G.; Jorns, M. S. *Biochemistry* **2002**, *41*, 9747–9750.

[9] Zhao, G.; Bruckner, R. C.; Jorns, M. S. *Biochemistry* **2008**, *47*, 9124–9135.

[10] Hassan-Abdallah, A.; Zhao, G.; Chen, Z.-W.; Mathews, F. S.; Schuman Jorns, M. *Biochemistry* **2008**, *47*, 2913–2922.

[11] Jorns, M. S.; Chen, Z.-w.; Mathews, F. S. *Biochemistry* **2010**, *49*, 3631–3639.

[12] Kommoju, P.-R.; Chen, Z.-w.; Bruckner, R. C.; Mathews, F. S.; Jorns, M. S. *Biochemistry* **2011**, *50*, 5521–5534.

[13] Hassan-Abdallah, A.; Zhao, G.; Jorns, M. S. *Biochemistry* **2006**, *45*, 9454–9462.

[14] Gadda, G. *Biochemistry* **2012**, *51*, 2662–2669.

[15] Chen, Z.; Trickey, P.; Jorns, M.; Mathews, F. Structure of the complex of monomeric sarcosine with its substrate analogue inhibitor 2-furoic acid at 1.3 A resolution.

[16] Cohen, J.; Arkhipov, A.; Braun, R.; Schulten, K. *Biophys. J.* **2006**, *91*, 1844–1857.

[17] Ayana, T.; Tokushi, S.; Kouhei, I.; Shunsuke, N.; Hirohiko, I.; Matthieu, C.; Fumihiro, K.; Sam-Yong, P.; Takayuki, T.; Takahisa, Y.; Shin-ya, K.; Shin-ichi, A. *Proc. Natl. Acad. Sci. USA* **2009**, *106*, 2612–2616.

[18] Moriguchi, T.; Ida, K.; Hikima, T.; Ueno, G.; Yamamoto, M.; Suzuki, H. *J. Biochem.* **2010**, *148*, 491–505.

[19] Chovancova, E.; Pavelka, A.; Benes, P.; Strnad, O.; Brezovsky, J.; Kozlikova, B.; Gora, A.; Sustr, V.; Klvana, M.; Medek, P.; Biedermannova, L.; Sochor, J.; Damborsky, J. *Pathways in Dynamic Protein Structures, PLoS Computational Biology 8: e1002708* **2012**,

[20] Ruscio, J. Z.; Kumar, D.; Shukla, M.; Prisant, M. G.; Murali, T. M.; Onufriev, A. V. *Proc. Natl. Acad. Sci. USA* **2008**, *105*, 9204–9209.

[21] Baron, R.; Binda, C.; Tortorici, M.; McCammon, J. A.; Mattevi, A. *Structure* **2011**, *19*, 212 – 220.

[22] Elber, R.; Karplus, M. *J. Am. Chem. Soc.* **1990**, *112*, 9161–9175.

[23] Laio, A.; Parrinello, M. *Proc. Natl. Acad. Sci. USA* **2002**, *99*, 12562–12566.

[24] Ceccarelli, M.; Anedda, R.; Casu, M.; Ruggerone, P. *Proteins-Struc. Func. & Bioinf.* **2008**, *71*, 1231–1236.

[25] Saam, J.; Ivanov, I.; Walther, M.; Holzhtter, H.-G.; Kuhn, H. *Proc. Natl. Acad. Sci. USA* **2007**, *104*, 13319–13324.

[26] Maragliano, L.; Vanden-Eijnden, E. *Chem. Phys. Lett.* **2006**, *426*, 168–175.

[27] Maragliano, L.; Vanden-Eijnden, E. *J. Chem. Phys.* **2008**, *128*, 1–10.

[28] Maragliano, L.; Fischer, A.; Vanden-Eijnden, E.; Ciccotti, G. *J. Chem. Phys.* **2006**, *125*.

[29] Maragliano, L.; Cottone, G.; Ciccotti, G.; Vanden-Eijnden, E. *J. Am. Chem. Soc.* **2010**, *132*, 1010–1017.

[30] Lapelosa, M.; Abrams, C. F. *J. Chem. Theory Comput.* **2013**, *9*, 1265–1271.

[31] Baron, R.; Riley, C.; Chenprakhon, P.; Thotsaporn, K.; Winter, R. T.; Alfieri, A.; Forneris, F.; van Berkel, W. J. H.; Chaiyen, P.; Fraaije, M. W.; Mattevi, A.; McCammon, J. A. *Proc. Natl. Acad. Sci. USA* **2009**, *106*, 10603–10608.

[32] Shadrina, M.; Peslherbe, G.; English, A. *Biochemistry* **2015**, *54*, 5279–5289.

[33] DiRusso, N.; Condurso, H.; Li, K.; Bruner, S.; Roitberg, A. *Chem. Sci.* **2015**, *6*, 6341–6348.

[34] Vanden-Eijnden, E.; Venturoli, M. *J. Chem. Phys.* **2009**, *130*.

[35] Weinan, E.; Weiqing, R.; Vanden-Eijnden, E. *Phys. Rev. B* **2002**, *66*, 052301.

[36] Jorgensen, W. L.; Chandrasekhar, J.; Madura, J. D.; Impey, R. W.; ; Klein, M. L. *J. Chem. Phys.* **1987**, *79*, 10.

[37] Simone, M.; Giovanni, C.; Holian,; Lee, B. *Mol. Phys.* **1993**, *78*, 533–544.

[38] Phillips, J. C.; Braun, R.; Wang, W.; Gumbart, J.; Tajkhorshid, E.; Villa, E.; Chipot, C.; Skeel, R. D.; Kale, L.; Schulten, K. *J. Comput. Chem.* **2005**, *26*, 1781–1802.

[39] MacKerell, Jr., A. D. et al. *J. Phys. Chem. B* **1998**, 3586–3616.

[40] MacKerell, Jr., A. D.; Banavali, N. K. *J. Comput. Chem.* **2000**, *21*, 105–120.

[41] Faradjian, A. K.; Elber, R. *J. Chem. Phys.* **2004**, *120*, 10880–10889.

[42] Mjek, P.; Elber, R. *J. Chem. Theory Comput.* **2010**, *6*, 1805–1817.

[43] Bucci, A.; Abrams, C. *J. Chem. Theory Comput.* **2014**, *10*, 2668–2676.

[44] Vanden-Eijnden, E.; Venturoli, M.; Ciccotti, G.; Elber, R. *J. Chem. Phys.* **2008**, *129*, 1–13.

[45] Vanden-Eijnden, E.; Venturoli, M. *J. Chem. Phys.* **2009**, *131*, 1–7.

[46] Dickson, A.; Warmflash, A.; Dinner, A. R. *J. Chem. Phys.* **2009**, *130*, 1–12.

[47] Held, M.; Metzner, P.; Noe, F. *Biophys. J.* **2011**, *100*, 701–710.

[48] Li, M. S.; Mai, B. K. *Curr. Bioinf.* **2012**, *7*, 342–351.

[49] Bonomi, M.; Branduardi, D.; Bussi, G.; Camilloni, C.; Provasi, D.; Raiteri, P.; Donadio, D.; Marinelli, F.; Pietrucci, F.; Broglia, R. A.; Parrinello, M. *Comput. Phys. Comm.* **2009**, *180*, 1961 – 1972.

[50] Humphrey, W.; Dalke, A.; Schulten, K. *J. Mol. Graph.* **1996**, *14*, 33–38.

[51] Abrams, C. F.; Vanden-Eijnden, E. *Proc. Natl. Acad. Sci. USA* **2010**, *107*, 4961–4966.

[52] Hassan-Abdallah, A.; Zhao, G.; Jorns, M. S. *Biochemistry* **2008**, *47*, 1136–1143.

[53] Towns, J.; Cockerill, T.; Dahan, M.; Foster, I.; Gaither, K.; Grimshaw, A.; Hazlewood, V.; Lathrop, S.; Lifka, D.; Peterson, G. D.; Roskies, R.; Scott, J. R.; Wilkin-Diehr, N. *Comput. in Sci. and Eng.* **2014**, *16*, 62–74.

[54] Baron, R.; McCammon, J. A.; Mattevi, A. *Curr. Op. in Struc. Biol.* **2009**, *19*, 672 – 679.

[55] Luty, B. A.; El Amrani, S.; Andrew, M. J. *J. Am. Chem. Soc.* **1993**, *115*, 11871–11877.

[56] Votapka, L. W.; Amaro, R. E. *PLOS Comp. Biol.* **2015**, *11*, 1–24.

[57] Bowman, G. R.; Geissler, P. L. *Proc. Natl. Acad. Sci. USA* **2012**, *109*, 11681–11686.

[58] Darve, E.; Rodriguez-Gomez, D.; Pohorille, A. *J. Chem. Phys.* **2008**, *128*, 1–13.

[59] Maragliano, L.; Vanden-Eijnden, E. *Chem. Phys. Lett.* **2007**, *446*, 182–190.

[60] Lapelosa, M.; Abrams, C. F. *Comput. Phys. Comm.* **2013**, *10*, 2310–2315.

[61] Metzner, P.; Schutte, C. *J. Chem. Phys.* **2006**, *125*, 1–17.

Appendix A. Sample configuration files and CV.inp

A.1 MD Configuration File

Sample MD configuration file for apo MSOX.

```
structure  msox.psf ;#File containing all bond information
coordinates si_apo.pdb ;#File containing structure coordinates

#Since restarting, don't need to specify temperature variable
#set temperature 310
#temperature $temperature

# Continue from First Nanosecond Files
set inputname      si_apo_h.restart
binCoordinates     $inputname.coor
binVelocities      $inputname.vel
extendedSystem     $inputname.xsc
firsttimestep      50000
numsteps           5000000

#New output name
set outputname     si_apo_h_2

#Parameter files needed to run simulation
paraTypeCharmm      on
parameters ../parameters/par_all27_prot_na.prm
parameters ../parameters/cfa_fadh.par
parameters ../parameters/toppar_all22_prot_heme.str

# These are specified by CHARMM
exclude             scaled1-4
1-4scaling          1.0

#Electrostatic cutoff
cutoff              10.0
switching           on
switchdist          8.5
pairlistdist        11.5

#Integrator parameters
timestep            2.0  ;#fs
rigidbonds      all  ;#Needed for 2fs timestep

#For periodic simulations
PME on
#NAMD decides grid spacing
pmegridspacing 2.0
```

```
#Cell origin and size for periodic cell
cellBasisVector1    83.7 0.0 0.0
cellBasisVector2    0.0 64 0.0
cellBasisVector3    0.0 0.0 67.5
cellOrigin          1.3900493383407593 1.2000447511672974 0
    .5536302328109741

wrapall on ;#Wrap all molecules in system

nonbondedFreq       1 ;#nonbonded forces every step
fullElectFrequency  2 ;#PME every 2 steps
stepspercycle       10 ;# Redo pairlists every 10 steps
outputName          $outputname

# Langevin Dynamics
langevin                on;             # do langevin dynamics
langevinDamping         5;               # damping coefficient (gamma
    ) of 5/ps
langevinTemp            310; #$temperature;    bath temperature
langevinHydrogen        no;              # don't couple langevin bath
     to hydrogens
seed                    98768674

#System pressue
langevinpiston          on
langevinpistontarget    1.013
langevinpistonperiod    200
langevinpistondecay     100
langevinpistontemp      310; #$temperature

#Constant pressure control
usegrouppressure        yes
useflexiblecell         no

#How often to output
restartfreq         10000
dcdfreq             10000
outputEnergies      1000

#Positional/Rotational Restraints
constraints on                          #Turn on harmonic restraints
consref si_apo.pdb    #PDB file containing positions of retrained
    atoms
consKFile apo.ref             #Reference PDB defining force constant
     values
conskcol B                  #Which column to look at for restraint
    force constant from above (in this case beta)

#How long to run
#minimize 1000
#run 50000
```

A.2 TAMD Configuration File

Sample TAMD simulation configuration file for apo-CB MSOX.

```
set home $env(HOME)

#Structure files are found in the CWD for this simulation.
coordinates             si_apo.pdb
structure               msox.psf
bincoordinates          si_apo_h.restart.coor
#binvelocities          foa_continue1.restart.vel
extendedSystem          si_apo_h.restart.xsc

#Parameter files are still stored in upper directories
paraTypeCharmm      on
parameters ../parameters/par_all27_prot_na.prm
parameters ../parameters/cfa_fadh.par
parameters ../parameters/toppar_all22_prot_heme.str
#parameters cfa_foa.par

set output              apo_kt3_3
outputname              $output
dcdfile                 ${output}.dcd
xstFile                 ${output}.xst
dcdfreq                 500
xstFreq                 1000

binaryoutput            yes
binaryrestart           yes
outputEnergies          100
restartfreq             1000

fixedAtoms              off

exclude                 scaled1-4
1-4scaling              1
COMmotion               no
dielectric              1.0

switching               on
switchdist              8.5
cutoff                  9
pairlistdist            10

set dt 2
firsttimestep           50000
numsteps                550000
timestep                $dt
stepspercycle           20
nonbondedFreq           2
fullElectFrequency      4
```

```
rigidbonds all

PME on
PMEgridspacing 1.0

wrapall on

set temperature         310
temperature             $temperature

langevin                on
langevinDamping         5
langevinTemp            $temperature
langevinHydrogen        no
seed                    987324

constraints on                      #Turn on harmonic restraints
consref si_apo.pdb    #PDB file containing positions of retrained
   atoms
consKFile apo.ref             #Reference PDB defining force constant
    values
conskcol B                  #Which column to look at for restraint
   force constant from above (in this case beta)


# Tcl forces for TAMD
# assumes you have cfacv installed in $HOME/cfacv
tclforces                       on
tclforcesscript                 ${home}/cfacv/cfacv_tclforces.tcl
#Coordinates to accelerate
set labelPDB                    02_beta1.pdb
#Defines collective variables
set cvINP                       cv.inp
#Restraints parameters
#set restrINP                   restr.inp
# optional: instead of naming 'restINP', you can have a single line
set restrPARAMS    {{k 200} {TAMDkT 3.0} {TAMDgamma 5.0} {TAMDdt 0
   .002}}
#TAMD output?
set TAMDof                      100

#run  50000
```

A.3 Force Evaluation Configuration File

Sample force calculation configuration file for apo-CB MSOX.

```
set home $env(HOME)
```

```
#Structure files are found in the CWD for this simulation.
coordinates             si_apo.pdb
structure               msox.psf
bincoordinates          frame00262.coor
#binvelocities          foa_continue1.restart.vel
extendedSystem          si_apo_h.restart.xsc

#Parameter files are still stored in upper directories
paraTypeCharmm      on
parameters ../parameters/par_all27_prot_na.prm
parameters ../parameters/cfa_fadh.par
parameters ../parameters/toppar_all22_prot_heme.str
#parameters cfa_foa.par

set output              force262
outputname              $output
dcdfile                 ${output}.dcd
xstFile                 ${output}.xst
dcdfreq                 50000
xstFreq                 10000

binaryoutput            yes
binaryrestart           yes
outputEnergies          100
restartfreq             1000

fixedAtoms              off

exclude                 scaled1-4
1-4scaling              1
COMmotion               no
dielectric              1.0

switching               on
switchdist              8.5
cutoff                  9
pairlistdist            10

set dt 2
firsttimestep           0
numsteps                500000
timestep                $dt
stepspercycle           20
nonbondedFreq           2
fullElectFrequency      4

rigidbonds all

PME on
PMEgridspacing 1.0

wrapall on
```

```
set temperature         310
temperature             $temperature

langevin                on
langevinDamping         5
langevinTemp            $temperature
langevinHydrogen        no
seed                    987324

constraints on                      #Turn on harmonic restraints
consref si_apo.pdb    #PDB file containing positions of retrained
   atoms
consKFile apo.ref             #Reference PDB defining force constant
    values
conskcol B                 #Which column to look at for restraint
   force constant from above (in this case beta)


# Tcl forces for TAMD
# assumes you have cfacv installed in $HOME/cfacv
tclforces                       on
tclforcesscript                 ${home}/cfacv/cfacv_tclforces.tcl
#Coordinates to accelerate
set labelPDB                    02_beta1.pdb
#Defines collective variables
set cvINP                       cv.inp
#Restraints parameters
#set restrINP                   restr.inp
# optional: instead of naming 'restINP', you can have a single line
set restrPARAMS    {{k 200} {TAMDkT 3.0} {TAMDgamma 0.0} {TAMDdt 0
   .002}}
#TAMD output?
set TAMDof                      100

#run  50000
```

A.4 Milestoning Configuration File

Sample milestoning configuration file for apo-CB MSOX.

```
set CFACV_BASEDIR $env(HOME)/cfacv
set home $env(HOME)

coordinates          si_apo.pdb
structure            msox.psf
set inputname       apo_cb_gate_2_pcen148.restart
binCoordinates      $inputname.coor
binVelocities       $inputname.vel
extendedSystem      $inputname.xsc
```

```
paraTypeCharmm      on
parameters ../parameters/par_all27_prot_na.prm
parameters ../parameters/cfa_fadh.par
parameters ../parameters/toppar_all22_prot_heme.str
#parameters cfa_foa.par


exclude             scaled1-4
1-4scaling          1.0
cutoff              9.0
switching           on
switchdist          8.5
pairlistdist        10.0

COMmotion               no
dielectric              1.0

wrapAll             on

PME                 on
PMEGridSpacing      1.0

firsttimestep       0
set dt 1
timestep            $dt
stepspercycle       20
nonbondedFreq       2
fullElectFrequency  4

rigidBonds          all

#set temperature     310
#temperature         310
langevin            on
langevinDamping     5
langevinTemp        310
langevinHydrogen    on
seed                987324

constraints on                      #Turn on harmonic restraints
consref si_apo.pdb    #PDB file containing positions of retrained
   atoms
consKFile apo.ref             #Reference PDB defining force constant
    values
conskcol B                 #Which column to look at for restraint
   force constant from above (in this case beta)


set output          apo_gate_10ns_pcen148
```

```
outputname          $output
dcdfile             ${output}.dcd
xstFile             ${output}.xst
dcdfreq             100000
xstFreq             100000
binaryoutput        yes
binaryrestart       yes
outputEnergies      10000
restartfreq         10000


tclforces                       on
tclforcesscript                 ${CFACV_BASEDIR}/
   cfacv_tclforces_mil_nonportal.tcl
set labelPDB                    O2_beta1.pdb
set cvINP                       cv.inp
set voronoi_centers_file       voronoi_apoCB_gate_pcen148.inp
# index of home cell; references indices in the
   $voronoi_centers_file
set home_cell           6
# number of steps since previous successful velocity reversal that
# we must wait before performing another one
set reverse_postwaitsteps       0
# indicate whether or not you want to save a configuration snapshot
# when a boundary is hit
set write_config_at_hit 0

run 5000000
```

A.5 CV.inp Input File

The collective variable file cv.inp for MSOX. It defines one CV, the COM of a group of tagged atoms.

```
CARTESIAN_X 0
CARTESIAN_Y 0
CARTESIAN_Z 0
```

Appendix B. Custom Single Sweep Scripts

B.1 RMSD Script

Script for determining the RMSD of MSOX from a loaded trajectory.

```
# Prints the RMSD of the protein atoms between each timestep
      # and the first timestep for the given molecule id (default:
          top)
      #
         ..........................................................


            puts "Start of Program: RMSD"

            #Define which molecule to operate on
            set mol "top"

            #Open a file to output data to
            set output [open 122012_foa_30ns_rmsd.dat w]

            # use frame 0 for the reference
            set reference [atomselect $mol "protein" frame 0]

            # the frame being compared
            set compare [atomselect $mol "protein"]

            #Get the number of timesteps
            set num_steps [molinfo $mol get numframes]

            puts "Going into loop"
            for {set frame 0} {$frame < $num_steps} {incr frame}
                {
                    puts "Calculating rmsd for frame $frame ..."
                    # get the correct frame
                    $compare frame $frame

                    # compute the transformation matrix
                    set trans_mat [measure fit $compare
                       $reference]

                    # do the alignment
                    $compare move $trans_mat

                    # compute the RMSD
                    set rmsd [measure rmsd $compare $reference]

                    # print the RMSD to the output file
                    puts $output "$frame    $rmsd"
```

```
}

close $output
```

B.2 COM O_2 Extraction Script

Script for extracting COM of O_2 from a loaded trajectory in VMD.

```
# Prints the center of mass and number of frames into an output file
#.....................................................................

puts "Start of Program: Get C.O.M."

#Define which molecule to operate on
set mol "top"

#Open a file to output data to
set output [open com.dat w]

#Get the number of timesteps
set num_steps [molinfo $mol get numframes]

#Output the number of steps
puts $output "$num_steps"

puts "Going into loop"
for {set frame 0} {$frame < $num_steps + 1} {incr
   frame} {
        #Choose the group to get the C.O.M. for
        set selection [atomselect $mol "resname O2"]

        puts "Calculating COM for frame: $frame"
        set COM [measure center $selection]
        puts $output "$COM"
}

close $output
```

B.3 Code to Ensure Centers Spacing is 2.5Å

FORTRAN code to check whether all centers have the desired spacing of 2.5Å. Compiles as gfortran CODENAME.f90 -o EXECUTABLENAME

```
!This program will read in a file of the form:
! [Blank Line]
! #Points
! X Y Z of C.O.M for O2 (in the case of MSOX)
```

```
!It will take all the data points and calculate the vector scalar
   length
!between each point and each other point and then loop over all
   points

program checkcenter

integer i, numframes, counter, j
double precision X,Y,Z,distance,cutoff
double precision,allocatable :: X_v(:), Y_v(:), Z_v(:)


!Open the file with the data
!.........................
open (unit= 2, file= "com_121012.dat")
open (unit= 3, file= "check.dat")
print*, "Files opened."
!Load the data. Store as three vectors so you can loop later
!.........................................................
 read(2,*) numframes
print*, "Number of frames to be read:",numframes

!Allocate vectors, initialize counter and define cutoff value
!.........................................................
 allocate( X_v(numframes), Y_v(numframes), Z_v(numframes))
 counter = 0
 cutoff = 2.5
!.........................................................

do i = 1, numframes
 read(2,*) X,Y,Z
 !print*, X,Y,Z
 X_v(i) = X
 Y_v(i) = Y
 Z_v(i) = Z
end do

!Calculate the scalar distance
do j = 1, numframes
 do i = 1, numframes
  distance = ((X_v(j) - X_v(i))**2 + (Y_v(j) - Y_v(i))**2 + (Z_v(j)
     - Z_v(i))**2)**0.5
  if (distance .ne. 0) then
   if (distance .lt. cutoff) then
    write(3,*) j,i,distance
    counter = counter + 1
   end if
  end if
 end do
end do

if (counter .gt. 0) then
```

```
 print*, "There are bad centers!"
 print*, "The number of bad centers is:", counter
end if
if (counter .eq. 0) then
 print*, "There are no bad centers!"
end if

!do i = 1,numframes
! write(3,*) X_v(i), Y_v(i), Z_v(i)
!end do

stop
end
```

B.4 Script for Extracting Z and θ Data from Logs

After force calculations are performed, this script will extract the necessary Z and θ values from the log files.

```
 set n 1
 puts $n
 puts "Entering Loop"
# i must be less than the number of log files to be read in.
#BE SURE TO SET BACK TO 0 WHEN FRAME 1 FINISHES!!!
for {set i 1} {$i < 356} {incr i} {
        puts "Reading frame$i.log"
### Open the log file for reading and the output .dat file for
   writing
        set file [open ../../../../paper/organized/apo_ob_logs/
           frame$n.log r]
        set output1 [open Z_apo_bo/Z$n.dat w]
        set output2 [open theta_apo_bo/theta$n.dat w]
#puts "Done Opening Files"
### Loop over all lines of the log file
        while { [gets $file line] != -1 } {
### Determine if a line contains Z or Th output. If so, write the
### timestep followed by the X Y Z components to the output file
        if {[lindex $line 1] == "Z"} {
                puts $output1 "[lindex $line 2] [lindex $line 3] [
                   lindex $line 4] [lindex $line 5]"
                }
        if {[lindex $line 1] == "Th"} {
                puts $output2 "[lindex $line 2] [lindex $line 3] [
                   lindex $line 4] [lindex $line 5]"
                }
        }
 #puts $counter
#puts "Output values to .dat"
### Close the log file and the output .dat file
        close $file
```

```
        close $output1
        close $output2
        incr n
}
```

B.5 Code for Calculating Running Averages from Force Calculations

FORTRAN code for calculating the running averages from the extracted Z and θ data. Compiles as gfortran CODENAME.f90 -o EXECUTABLENAME

```
 program average_force
integer i, stat, B, frame
double precision averageX, averageY, averageZ, z_x, z_y, z_z, th_x,
   th_y, th_z
double precision c_x,c_y,c_z
double precision n,k,m,l
 character(len=1) :: ST1
 character(len=2) :: ST2
 character(len=3) :: ST3

!First line of the file is number of frames to read
!uncomment the coded stuff to get it back to normal
!.............................................
 !open (unit= 6, file= "../com_data/com_newcenters.dat")
 !read(6,*) frame
 !close (6)
  frame = 355
  print*, frame

!Numbering scheme begins with frame 0
! 1 in the case of bound system for msox
!.................................
B = 1

!If B (a counter) is greater than the number of frames end program
!
   ..................................................................

200 if ( B .gt. frame ) then
 go to 100
end if

!Convert an integer to a string using internal files
!write to ST1 with a format of (I1.0)(1 place with no leading zeroes
   ) the value B
!................................................
write(UNIT=ST1, FMT='(I1.0)') B
if (B .gt. 9) then
 write(UNIT=ST2, FMT='(I2.0)') B
end if
```

```
if (B .gt. 99) then
 write(UNIT=ST3, FMT='(I3.0)') B
end if


!Zero is not an integer so you need an initial string
if ( B .eq. 0) then
 ST1 = '0'
end if


!Be sure the string and B values are identical
if (B .lt. 9) then
 print*, ST1, B
end if
if (B .gt. 9) then
 print*, ST2, B
end if
if (B .gt. 99) then
 print*, ST3, B
end if

!Open the files with the data
!Open output files
!.........................
!open (unit= 7, file= "../com_data/com_newcenters_friendly.dat")
open (unit= 2, file= "Z_apo_bo/Z" // ST1 // ".dat")
open (unit= 3, file= "theta_apo_bo/theta" // ST1 // ".dat")
open (unit= 4, file= "average_apo_bo/average" // ST1 // ".dat")
open (unit= 5, file= "centers_forces_apo_bo.dat")
if (B .gt. 9) then
 open (unit= 2, file= "Z_apo_bo/Z" // ST2 // ".dat")
 open (unit= 3, file= "theta_apo_bo/theta" // ST2 // ".dat")
 open (unit= 4, file= "average_apo_bo/average" // ST2 // ".dat")
end if

if (B .gt. 99) then
 open (unit= 2, file= "Z_apo_bo/Z" // ST3 // ".dat")
 open (unit= 3, file= "theta_apo_bo/theta" // ST3 // ".dat")
 open (unit= 4, file= "average_apo_bo/average" // ST3 // ".dat")
end if

!Ask for spring constant
!print*, "Spring Constant Value?"
!read(*,*) k
!For now hard code 1
!...................
k = 200

!initialize averages as 0
!.......................
averageX = 0
```

```
averageY = 0
averageZ = 0

!Calculate the average
!.....................

!Outputs the current COM as well
!read(7,*) c_x,c_y,c_z
do i = 1,5000000

  !Here use IOSTAT to determine if end of file has been reached
  ! stat = 0 == All good            stat < 0 == end of file
 read(2,*,IOSTAT=stat) l, z_x, z_y, z_z
 read(3,*,IOSTAT=stat) m, th_x, th_y, th_z
 !print*, z_x, th_x

 !n = 5000
if (stat .eq. 0) then
! compute running average (up to current time point)
 averageX = averageX + (k)*(th_x - z_x)
 averageY = averageY + (k)*(th_y - z_y)
 averageZ = averageZ + (k)*(th_z - z_z)
 write(4,*) l/500000,averageX/i,averageY/i,averageZ/i
end if
if (stat .lt. 0) then
 !Output final running average value before going to next file
 !in format CenterNum position[X Y Z] force[X Y Z]
 write(5,*) B,z_x,z_y,z_z,averageX/i,averageY/i,averageZ/i
 B = B + 1
 print*, B,frame,i
 go to 200
end if
end do

100 stop
end
```

Appendix C. Milestoning Codes and Script

C.1 Codes and Scripts for Setting up Milestoning Simulations

C.1.1 Code to Identify Frame Nearest String Image

A C code which calculates the distance from each point in a list of COM of O_2 positions to a specific image on a string. It outputs the frame containing the minimum distance to each string image. Compiles as gcc FILENAME.c -lm -o NameYouLike

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

/*Reads in 2 files and computes cartesian distance between*/
/*String image in format (Image X Y Z) and molecule COM in format (
   Frame X Y Z) to compare against acceptance criterion*/
/* compiles as gcc find_frames_gen_2.c -lm -o {Name You Like}*/


main (int argc, char * argv[]) {



int i;
double str_X,str_Y,str_Z;
double o2_X, o2_Y, o2_Z;
int fr, image, crit_frame;
double crit2 = 1;
double mag_dist_old;
double distance, mag_dist;
char im[5], string_in[255], com_in[255];


/*At run time define input files at minimum, crit is set to 1AA by
   default*/
for (i=1;i<argc;i++) {
    if (!strcmp(argv[i],"-crit")) {
      sscanf(argv[++i],"%lf",&crit2);
    }
    else if (!strcmp(argv[i],"-string")) {
      sscanf(argv[++i],"%s",&string_in);
    }
    else if (!strcmp(argv[i],"-com")) {
      sscanf(argv[++i],"%s",&com_in);
    }
  }
/*Output file name*/
```

```
FILE *out_file = fopen("ms_bound_gate_frame.txt", "w"); // write
   only

/*Input file 1 (string)*/
FILE *in_file1;
char *mode = "r";
in_file1 = fopen(string_in, mode);
if (in_file1 == NULL) {
        printf("No String image data supplied.\n");  //If input file
            does not exist, exit program
        exit (0);
}

/*Note fscanf requires EXACT formating*/
/*Read in string value in outer loop*/
float a = 0, b = 0, c = 0, d = 0, o2x = 0, o2y = 0, o2z = 0, frame =
    0, crit = crit2;

while (fscanf(in_file1, "%f %f %f %f\n", &a, &b, &c, &d) == 4) {
mag_dist_old = crit;
crit_frame = 0;
 image = a + 1;
 str_X = b;
 str_Y = c;
 str_Z = d;
printf(" Finding frame for String Image: %d\n", image);
        /*Read O2 COM file*/
        FILE *in_file2;
        char *mode = "r";
        in_file2 = fopen(com_in, mode);
        if (in_file2 == NULL) {
        printf("No COM data supplied.\n");  //If input file does not
            exist, exit program
        exit (0);
        }
        /*Begin loop for calculation of distance*/
        /*read in first frame o2 com*/

        while (fscanf(in_file2, "%f %f %f %f\n", &frame, &o2x, &o2y,
            &o2z) == 4) {
                fr = frame;
                o2_X = o2x;
                o2_Y = o2y;
                o2_Z = o2z;

        /*Calculate distance*/
        distance = (o2_X - str_X)*(o2_X - str_X) + (o2_Y - str_Y)*(
           o2_Y - str_Y) + (o2_Z - str_Z)*(o2_Z - str_Z);
        mag_dist = pow(distance,0.5);

        /*Check distance criterion*/
        if (mag_dist <= crit){
```

```
          /*Looking for minimum distance!*/
          /*Mag_dist_old is originally defined as crit*/
          /*This ensures the first acceptance will define a new
             mag_dist_old*/
          if (mag_dist_old >= mag_dist){
                mag_dist_old = mag_dist; //update mag_dist
                crit_frame = fr; //update frame
          }
        }

        }

        fprintf(out_file, "%i %i %.3lf\n",image, crit_frame,
           mag_dist_old); // write to file
        fclose(in_file2);

        /*image+= 1;*/
}
fclose(out_file);
}
```

C.1.2 Script for Extracting the Correct Frame for Each String Image

Script which extracts from the loaded trajectory the correct frame that will be used in milestoning.

```
#Generate .coor files for initial configurations for
    Milestoning calcs
#Assumes composite DCD is loaded as TOP molecule
#
   ************************************************************



#Open the file
set file [open ms_frames_midlys_15to32.txt r]
#Read all lines in the file
while { [gets $file line] != -1 } {
#Get data on line
set im [lindex $line 0]
set frame [lindex $line 1]
set distance [lindex $line 2]

#Define frame to output
set TAMDframe "top"
set all_atoms [atomselect $TAMDframe "all" frame
   $frame]
$all_atoms writenamdbin  mstone_midlys_cell${im}
   .coor
 }
```

```
                puts "Done!"
```

C.1.3 Script to Load All Frames Extracted

A simple TcL script to load all frames generated.

```
for { set i 15 } { $i < 85 } { incr i } {

  mol addfile mstone_midlys_cell$i.coor

}
```

C.1.4 Script to Generate Input Files for Milestoning

A TcL script which moves the O_2 to the center of each cell, then generates input structure files for minimization.

```
puts "Start of Program: Move O2 and Output Sys Center"
                #****************************************
                #*****Only info to change between runs*****

                #String pdb
                set file [open ~/ocd_everything/msox/strings/APO/
                   closed_bridge/midlys_apocb_initMS.pdb r]
                #Output
                set output [open center_midlys.dat w]
                #Number of string images
                set num_images 30

                #Location of system
                set sys "top"

                #Molecule you want to move
                set molecule "resname O2"
                #****************************************
                #****************************************

                set c 1
                set str_image 1

                #Loop over all lines in specified PDB
                while { [gets $file line] != -1 } {
                #puts "$line"
                set cell [expr {$str_image - 1}]
                puts "Cell $cell Str_image $str_image"

                if {$cell == 0} {
                puts "Cell 0 no file generation!"
```

```
set X_str1 [lindex $line 6]
set Y_str1 [lindex $line 7]
set Z_str1 [lindex $line 8]

} elseif { [expr {$str_image % 2}] == 0} {
set X_str1 [lindex $line 6]
set Y_str1 [lindex $line 7]
set Z_str1 [lindex $line 8]

} else {

set X_str1 [lindex $line 6]
set Y_str1 [lindex $line 7]
set Z_str1 [lindex $line 8]
}
puts "String COM $X_str1 $Y_str1 $Z_str1"
if { $cell < $num_images } {
puts "Generating Cell $cell Files"
#Get XYZ components of string imageall
#Store as Pt1 (on odd iterations) and Pt2 (on even
   iterations) respectively
#On even iterations, do lots of stuff

#Calculate O2 COM
#NOTE THE MINIMUM DISTANCE FRAMES MUST BE LOADED
   INTO VMD
#AND CORRESPOND TO APPROPRIATE CELL
set o2_com [atomselect $sys "$molecule" frame $cell]
set o2_pdb [atomselect $sys "$molecule" frame $cell]
set com [measure center $o2_com]
puts "O2 old: $com"

#set midX [expr {($X_str2 + $X_str1)/2}]
#set midY [expr {($Y_str2 + $Y_str1)/2}]
#set midZ [expr {($Z_str2 + $Z_str1)/2}]
#puts "midxyz: $midX $midY $midZ"
#Define moveby Vectors
#
   ************************************************************

set mvX [expr {$X_str1 - [lindex $com 0]}]
set mvY [expr {$Y_str1 - [lindex $com 1]}]
set mvZ [expr {$Z_str1 - [lindex $com 2]}]
set vec [list $mvX $mvY $mvZ]
puts "moveby vec: $vec"

#TO USE MOVEBY WITH A VECTOR AS A VARIABLE YOU MUST
   DEFINE A LIST AS SHOWN ABOVE
#
   ************************************************************
```

```
#In vmd the O2 moves to the correct location.
   However, the output is
# incorrect.
#Move o2 COM
$o2_com moveby $vec
set o2com [measure center [atomselect $sys "resname
   O2"]]
puts "new o2com: $o2com"

#Output minimization PDB
set all [atomselect $sys "all" frame $cell]
$all writepdb mstone_midlys_min$cell.pdb
#Output fixed atoms pdb
$all set beta 0
$o2_pdb set beta 1
$all writepdb o2_fix_midlyscell$cell.pdb

#Output system center (for now assume periodic cube
   will be ~ constant)
set center [measure center $all]
puts $output "$center"
}
set str_image [expr {$str_image + 1}]
}

close $output
```

C.1.5 Script to Generate Minimization Files

A bash script which will generate the necessary minimization files. Recall that we moved the O_2 so that minimizing will allow nearby residues to relax and not be in improper locations.

```
#!/bin/bash

#Loop over the number of files you want. MAKE SURE TO CHANGE THIS!
for (( c=0; c<=10; c++))

do

#Define replacement variables
#Here the numbering for the frames is inconvenient so I had to use
   this awkward 3 variable approach
     pdb="mstone_midlys_min$c.pdb"
     fix="o2_fix_midlyscell$c.pdb"
     output="mil_min_midlyscell$c"
#For The template file copy it and rename .old (so as not to destroy
    the good template)
     for fl in mil_min_template.conf; do
     cp $fl $fl.old
```

```
#Check for patterns and replace with replacement variables. Then
    pipe it to a new file that is numbered according to the loop
    index
      sed  -e "s/mstone_cell1.pdb/$pdb/g" -e "s/o2_fix_cell1.pdb/$fix
         /g" -e "s/mil_min_cell1/$output/g" $fl.old >
         midlys_min_cell$c.conf

      done
done
```

C.1.6 Script to Run Minimizations Locally

A bash script to run all the minimizations locally.

```
#!/bin/bash

#Loop over the number of files you want. MAKE SURE TO CHANGE THIS!
for (( c=1; c<=10; c++))

do

      #Run namd minimization scripts
      /home/ajb/NAMD_2.9_Linux-x86_64-multicore/namd2 +idlepoll +p14
         midlys_min_cell$c.conf > mil_min_midlyscell$c.log

done
```

C.1.7 Script to Generate Submission Files for Stampede

A bash script to generate the submission files for stampede.

```
#!/bin/bash

#Loop over the number of files you want. MAKE SURE TO CHANGE THIS!
for (( c=164; c<=170; c++))

do

#Define replacement variables
#Here the numbering for the frames is inconvenient so I had to use
    this awkward 3 variable approach
      enote="midlys_cell$c"
      conf="apo_midlys_cell$c.conf"
      output="apo_midlys_5ns_cell$c.log"
#For The template file (frame1.conf here) copy it and rename .old (
    so as not to destroy the good template)
      for fl in stampede_ms_template; do
      cp $fl $fl.old
```

```
#Check for patterns and replace with replacement variables. Then
   pipe it to a new file that is numbered according to the loop
   index
     sed  -e "s/stampede_dev_mil_30/$enote/g" -e "s/
        apo_cb_sarc_cell30.conf/$conf/g" -e "s/
        apo_cb_sarc_5ns_cell30.log/$output/g" $fl.old >
        stampede_apoCB_midlys_cell$c

     done
done
```

C.1.8 Script to Generate Configuration Files for Stampede

A bash script to generate the configuration files for stampede.

```
#!/bin/bash

#Loop over the number of files you want. MAKE SURE TO CHANGE THIS!
for (( c=164; c<=170; c++))

do

#Define replacement variables
#Here the numbering for the frames is inconvenient so I had to use
   this awkward 3 variable approach
     cell="$(expr $c - 164)"
     coor="mil_min_midlyscell$cell.coor"
     xsc="mil_min_midlyscell$cell.xsc"
     output="apo_midlys_cell$c"
     voronoi="voronoi_apoCB_midlys.inp"

#For The template file (frame1.conf here) copy it and rename .old (
   so as not to destroy the good template)
     for fl in milestoning_template.conf; do
     cp $fl $fl.old
#Check for patterns and replace with replacement variables. Then
   pipe it to a new file that is numbered according to the loop
   index
     sed  -e "s/ mil_min_cell30.coor/$coor/g" -e "s/
        mil_min_cell30.xsc/$xsc/g" -e "s/apo_cb_sarc_cell30/$output/
        g" -e "s/30_cell/$c/g" -e "s/ voronoi_apoCB_sarc.inp/
        $voronoi/g" $fl.old > apo_midlys_cell$c.conf

     done
done
```

C.2 Codes and Scripts for Analysis of Milestoning Data

C.2.1 Script to Extract Transitions from Log Files

A TcL script to extract the transition data from individual log files from milestoning.

```
puts "Start of Program: Extract Milestoning Data"
#****************************************
if {$argc > 0} {puts "The other arguments are: $argv
   " }
#Data File
set file [open $argv.log r]
#Output file and define distance criterion in
   angstroms
set output1 [open $argv.dat w]
puts $argv
puts "Required information defined."
#*********************************************

#Loop over all lines in data file
while { [gets $file line] != -1 } {
 # [lindex $line 3]
 if {[lindex $line 3] == "attempted"} {
 #puts $output1 "$line"
 puts $output1 "[lindex $line 2] [lindex $line 6] [
    lindex $line 8]"

 }

}

puts "DONE!"
close $output1
```

C.2.2 Script to Extract All Boundary Violations from Log Files

A TcL script to extract all boundary violation data from individual log files from milestoning.

```
#puts "Start of Program: Extract Hitting Data"
#****************************************
if {$argc > 0} {puts "The other arguments are: $argv
   " }
#Data File
set file [open $argv.log r]
#Output file and define distance criterion in
   angstroms
set output1 [open hitting_$argv.dat w]
puts $argv
```

```
puts "Required information defined."
#*****************************************

#Loop over all lines in data file
while { [gets $file line] != -1 } {
 # [lindex $line 3]
 if {[lindex $line 0] == "CFACV/MIL/C)" } {
        set timestep [lindex $line 1]
        #puts $timestep
        set expression [expr $timestep % 200]
        #puts $expression
        if { $expression != 0 } {
         puts $output1 "[lindex $line 1] [lindex
            $line 3] [lindex $line 5] [lindex $line
            7]"
        }

 }

}

#puts "DONE!"
close $output1
```

C.2.3 Script to Visualize Boundary Violations

A TcL script which draws the hitting data as different colored points in VMD.

```
#puts "Start of Program: Visualize Data"
#*****************************************
set colorcounter 0
#for {set c 1} {$c < 75} {incr c} {
#Data File
set actual [expr $c + $c + 1]
set file [open allhits.dat r]

#*****************************************

#Loop over all lines in data file
while { [gets $file line] != -1 } {

set X1 [lindex $line 1]
set Y1 [lindex $line 2]
set Z1 [lindex $line 3]
set vec1 [list $X1 $Y1 $Z1]
if {$colorcounter == 0} {
draw color red
draw point $vec1
}
if {$colorcounter == 1} {
draw color blue
```

```
draw point $vec1
}
if {$colorcounter == 2} {
draw color green
draw point $vec1
}
if {$colorcounter == 3} {
draw color purple
draw point $vec1
}

}

puts "old cc $colorcounter"
puts "incrementing colorcounter"
set colorcounter [expr $colorcounter + 1]
puts "new cc $colorcounter"

if {$colorcounter == 4} {
set colorcounter 0
}
}
```

C.2.4 Code to Calculate Average Transition Time

A C code which calculates the average time per transition from the transition data, and can be plotted using GNUPLOT afterwards.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

/*Reads in master cell center file and write a lookup table*/


/* compiles as gcc mapping.c -lm -o {Name You Like}*/


main (int argc, char * argv[]) {


int i;
double time, time_old, cell_in, cell_to, cell_to_old, av_numer;
double time_st;
double average;
char ifile[255];


/*At run time define input. Need to define input file (
   time_cell**.dat) */
for (i=1;i<argc;i++) {
```

```
    if (!strcmp(argv[i],"-ifile")) {
      sscanf(argv[++i],"%s",&ifile);
    }

  }

/*Input file 1 (master cell)*/
FILE *in_file1;

char *mode = "r";
in_file1 = fopen(ifile, mode);
if (in_file1 == NULL) {
        printf("No data available for average.\n");  //If input file
            does not exist, exit program
        exit (0);
}

/*Note fscanf requires EXACT formating*/
/*Read input file and average*/
float a = 0, b = 0, c = 0, d = 0;
int av_count=0;
int num_transitions = 1;
double co;
while (fscanf(in_file1, "%f %f %f\n", &a, &b, &c) == 3) {

 time = a;
 cell_in = b;
 cell_to = c;
 /*printf("%.3lf %.3lf %.3lf\n",time, cell_in, cell_to);*/


 /* Averaging Block*/

/*printf("%i %.3lf %.3lf\n",av_count, cell_to_old, cell_to);*/
 /*A transition occurs if cell_to(old) not equal to cell_to(new)*/
 if (av_count != 0 && cell_to != cell_to_old){
        if (time < time_old){
                time = time + 5000000;
                co = co + 1;
        }
        /*printf("inside av_count != 1\n");*/
        /*Calculate numerator for the average*/
        /*time - time_old = time to transition between milestones*/
        time_st = time - time_old;
        if (time_st > 0){
        av_numer = av_numer + (time - time_old);
        /*printf("%.3lf %.3lf\n", cell_to, cell_to_old);*/
        /*printf("%.3lf %.3lf\n", time_old, time);*/
        /*Store current time*/

        time_old = time;
        /*printf("%.3lf\n", av_numer);*/
```

```
        /*increment av_count*/
        av_count += 1;
        num_transitions += 1;
        cell_to_old = cell_to;

        average = av_numer / (av_count - 1);
        printf("%i %.6lf %.6lf\n", num_transitions, average, time_st
           );
        }
        }

if (av_count == 0){
    /*printf("inside av_count ==1\n");*/
   /*Store first cell_to*/
   cell_to_old = cell_to;
   /*Store first time*/
   time_old = time;
   av_count += 1;

 }

 }
}
```

C.2.5 Code to Map Data from a Stable Tesselation

A C code which maps a stable tesselation to a more user friendly numbering system. This code cannot distinguish sphereical shell hits however.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

/*Reads in master cell center file and write a lookup table*/


/* compiles as gcc mapping.c -lm -o {Name You Like}*/


main (int argc, char * argv[]) {


int i, min_i, check;
double str_X,str_Y,str_Z;
double o2_X, o2_Y, o2_Z, old;
int fr, image, crit_frame;
int home;
double mag_dist_old;
double distance, mag_dist;
char im[5], master_in[255], com_in[255];
```

```
double ind[300], X[300], Y[300], Z[300];


/*At run time define input. Need to define home cell (right now),
   master cell files, and hitting file to convert */
for (i=1;i<argc;i++) {
    if (!strcmp(argv[i],"-home")) {
      sscanf(argv[++i],"%i",&home);
    }
    else if (!strcmp(argv[i],"-master")) {
      sscanf(argv[++i],"%s",&master_in);
    }
    else if (!strcmp(argv[i],"-hitting")) {
      sscanf(argv[++i],"%s",&com_in);
    }
  }
/*Output file name*/
/*FILE *out_file = fopen("time_cell""home".txt", "w"); // write only
    */

/*Input file 1 (master cell)*/
FILE *in_file1;
FILE *in_file2;
char *mode = "r";
in_file1 = fopen(master_in, mode);
if (in_file1 == NULL) {
        printf("No cell center data available.\n");  //If input file
            does not exist, exit program
        exit (0);
}
in_file2 = fopen(com_in, mode);
        if (in_file2 == NULL) {
        printf("No hitting data supplied.\n");  //If input file does
            not exist, exit program
        exit (0);
        }

/*Note fscanf requires EXACT formating*/
/*Read in all cell centers and develop lookup table*/
float a = 0, b = 0, c = 0, d = 0;
int index=1;
while (fscanf(in_file1, "%f %f %f %f\n", &a, &b, &c, &d) == 4) {
 /*printf("%.0lf %.3lf %.3lf %.3lf\n", a, b, c, d);*/
 X[index] = b;
 Y[index] = c;
 Z[index] = d;
 index = index + 1;

}


/*printf("Lookup table complete\n");*/
```

```
float h_x, h_y, h_z, ts;
int step;
double hit_x, hit_y, hit_z;
/*Read in hitting file and determine which cell it maps to*/
while (fscanf(in_file2, "%f %f %f %f\n", &ts, &h_x, &h_y, &h_z) ==
   4) {

 old = 500;
 /*Store hitting value for operation*/
 hit_x = h_x;
 hit_y = h_y;
 hit_z = h_z;
 step = ts;

 /*Check against cell table to determine which cell it escapes to*/

/*Distance check*/
/*Right now it only loops to index before last. This assumes last
   index is ficititous and added*/
/*to account for a portal that has overlapping cells, and a
   different point was used */
/*to signify the sphere center*/
for (i=1;i<index-1;i++){
distance = (hit_x - X[i])*(hit_x - X[i]) + (hit_y - Y[i])*(hit_y - Y
   [i]) + (hit_z - Z[i])*(hit_z - Z[i]);
        mag_dist = pow(distance,0.5);

/*Check distance criterion*/
        if (mag_dist <= old && i != home + 1 && i != index){
          /*Looking for minimum distance!*/
          /*old is originally defined as some ridiculous number*/
          /*This ensures the first acceptance will define a new old*
             /
                old = mag_dist; //update mag_dist
                min_i = i; //update index corresponding to minimum
                   value
                check = min_i - 1;
         }

        /*if (mag_dist <= old && home == check){
                min_i = min_i - 1;
        } */
}
printf("%i %i %i\n", step, home, min_i - 1);
 }
}
```

C.2.6 Code to Map Data from Solvent Sphereical Shell

A C code which maps a stable tesselation to a more user friendly numbering system. This version was used to map sphereical shell hits.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

/*Reads in master cell center file and write a lookup table*/


/* compiles as gcc mapping.c -lm -o {Name You Like}*/


main (int argc, char * argv[]) {


int i, min_i, check;
double str_X,str_Y,str_Z;
double o2_X, o2_Y, o2_Z, old;
int fr, image, crit_frame;
int home, sphere_center;
double mag_dist_old;
double distance, mag_dist;
char im[5], master_in[255], com_in[255];
double ind[300], X[300], Y[300], Z[300];


/*At run time define input. Need to define home cell (right now),
   master cell files, and hitting file to convert */
for (i=1;i<argc;i++) {
    if (!strcmp(argv[i],"-home")) {
      sscanf(argv[++i],"%i",&home);
    }
    else if (!strcmp(argv[i],"-master")) {
      sscanf(argv[++i],"%s",&master_in);
    }
    else if (!strcmp(argv[i],"-spcen")) {
      sscanf(argv[++i],"%i",&sphere_center);
    }
    else if (!strcmp(argv[i],"-hitting")) {
      sscanf(argv[++i],"%s",&com_in);
    }
  }
/*Output file name*/
/*FILE *out_file = fopen("time_cell""home".txt", "w"); // write only
    */

/*Input file 1 (master cell)*/
FILE *in_file1;
FILE *in_file2;
char *mode = "r";
in_file1 = fopen(master_in, mode);
if (in_file1 == NULL) {
        printf("No cell center data available.\n");  //If input file
            does not exist, exit program
```

```
        exit (0);
}
in_file2 = fopen(com_in, mode);
        if (in_file2 == NULL) {
        printf("No hitting data supplied.\n");  //If input file does
            not exist, exit program
        exit (0);
        }

/*Note fscanf requires EXACT formating*/
/*Read in all cell centers and develop lookup table*/
float a = 0, b = 0, c = 0, d = 0;
int index=1;
while (fscanf(in_file1, "%f %f %f %f\n", &a, &b, &c, &d) == 4) {
 /*printf("%.0lf %.3lf %.3lf %.3lf\n", a, b, c, d);*/
 X[index] = b;
 Y[index] = c;
 Z[index] = d;
 index = index + 1;
 /*printf("%i\n", index);*/
}


/*printf("Lookup table complete\n");*/
float h_x, h_y, h_z, ts;
int step;
double hit_x, hit_y, hit_z;
/*Read in hitting file and determine which cell it maps to*/
while (fscanf(in_file2, "%f %f %f %f\n", &ts, &h_x, &h_y, &h_z) ==
   4) {

 old = 500;
 /*Store hitting value for operation*/
 hit_x = h_x;
 hit_y = h_y;
 hit_z = h_z;
 step = ts;

 /*Check against cell table to determine which cell it escapes to*/
int sphere;
double ch_dist,mag_portal;
sphere = sphere_center + 1;
/*sphere = 90;*/

/*Distance check*/
/*see mapping.c for explanation of why i<index and not i<=index*/
for (i=1;i<index-1;i++){
ch_dist = (hit_x - X[sphere])*(hit_x - X[sphere]) + (hit_y - Y[
   sphere])*(hit_y - Y[sphere]) + (hit_z - Z[sphere])*(hit_z - Z[
   sphere]);
        mag_portal = pow(ch_dist,0.5);
```

```
distance = (hit_x - X[i])*(hit_x - X[i]) + (hit_y - Y[i])*(hit_y - Y
   [i]) + (hit_z - Z[i])*(hit_z - Z[i]);
        mag_dist = pow(distance,0.5);

/*Check distance criterion*/
/*For portal cells only, if ch_dist is greater than sphere radius
define the transition cell as -1 to denote solvent*/
/*printf("mag_portal %.3lf  mag dist %.3lf\n", mag_portal, mag_dist)
   ;*/
        if (mag_portal >= 12.5){
                min_i = 0;
        }
        else if (mag_dist <= old && i != home + 1 && index != i){
          /*Looking for minimum distance!*/
          /*old is originally defined as some ridiculous number*/
          /*This ensures the first acceptance will define a new old*
             /
                old = mag_dist; //update mag_dist
                min_i = i; //update index corresponding to minimum
                   value
                check = min_i - 1;
         }

        /*if (mag_dist <= old && home == check){
                min_i = min_i - 1;
        } */
}
printf("%i %i %i\n", step, home, min_i - 1);
 }
}
```

C.2.7 Script to Find Shell Hitting

A simple bash script which checks each time cell for sphereical boundary hitting. This helps find the input to the MFPT code.

```
#!/bin/bash

#Loop over the number of files you want. MAKE SURE TO CHANGE THIS!
#chmod 744 before you use this file

for (( c=0; c<=300; c++))
do
        if [ -a ../../cell_data/time_cell$c.dat ]
          then
                grep "$c -1" ../../cell_data/time_cell$c.dat >>
                   sphere_cells_apo.dat
        fi

done
```

C.2.8 Script to Cull Data

A TcL script which can remove fictitious hitting from a time cell file, and replace it with an imposed boundary condition.

```
puts "Start of Program: Remove Data from
   time_cell*.dat"
#****************************************
if {$argc > 0} {puts "The other arguments are: $argv
   " }
#Data File
set file [open $argv.dat r]
#Output file and define distance criterion in
   angstroms
set output1 [open culled_$argv.txt w]
puts $argv
puts "Required information defined."
#****************************************

#Loop over all lines in data file
while { [gets $file line] != -1 } {

# [lindex $line 3]
#CHANGE THIS TO YOUR FICTITIOUS HITTING BOUNDARY
 if {[lindex $line 2] == "22" || [lindex $line 2] ==
     "23" || [lindex $line 2] == "24" || [lindex
    $line 2] == "25" || [lindex $line 2] == "26"} {
**************************************************

 #puts $output1 "$line"
 puts $output1 "[lindex $line 0] [lindex $line 1] -2
    "

 } else {
 puts $output1 "$line"
 }

}

puts "DONE!"
close $output1
```

C.2.9 Code to Adjust Numbering of Time Cell Data

A C code that modifies existing time cell data. With multiple segments, the MFPT code will generate errors if the time step numbering is incorrect. This code will ensure numbering is correct.

```
#include <stdio.h>
#include <stdlib.h>
```

```
#include <math.h>

/*Reads in master cell center file and write a lookup table*/


/* compiles as gcc mapping.c -lm -o {Name You Like}*/


main (int argc, char * argv[]) {


int i;
double time, time_old, cell_in, cell_to, cell_to_old, av_numer;
double time_st;
double average;
char ifile[255];


/*At run time define input. Need to define input file (
   time_cell**.dat) */
for (i=1;i<argc;i++) {
    if (!strcmp(argv[i],"-ifile")) {
      sscanf(argv[++i],"%s",&ifile);
    }

  }

/*Input file 1 (master cell)*/
FILE *in_file1;

char *mode = "r";
in_file1 = fopen(ifile, mode);
if (in_file1 == NULL) {
        printf("No data available for average.\n");  //If input file
            does not exist, exit program
        exit (0);
}

/*Note fscanf requires EXACT formating*/
/*Read input file and average*/
float a = 0, b = 0, c = 0, d = 0;
int av_count=0;
int num_transitions = 1;
double co;
time_old = 0;
co = 0;
while (fscanf(in_file1, "%f %f %f\n", &a, &b, &c) == 3) {

 time = a;
 cell_in = b;
 cell_to = c;
```

```
        if (time < time_old && time_old != 0){
                co = co + 1;
        }

        time_old = time;
                printf("%i %.6lf %.6lf\n", time + 5000000*co,
                   cell_in, cell_to);

        }


}
```

C.2.10 Sample Voronoi.inp Input File

The following is a subset of a larger voronoi.inp file. In it, we illustrate the numbering convention as well as how to initialize a spherical shell boundary. Sections of different strings are also noted. This data cannot be used as exhibited here, as the format for any voronoi.inp file must be "Index X Y Z".

```
**Gate String Originating at the Active Site Minimum**
**Note that numbering begins at zero, and not one**
**Format: ‘‘Cell StringIndex X Y Z’’
0 2 -7.118 2.088 -1.895
1 3 -6.687 1.217 -1.971
2 4 -6.381 0.293 -1.905
3 5 -6.575 -0.537 -1.431
4 8 -7.264 -2.204 0.753
5 11 -6.061 -2.263 3.383
6 12 -5.327 -2.237 4.024
7 13 -4.422 -2.289 4.383
8 14 -3.463 -2.378 4.535
9 15 -2.492 -2.444 4.598
.
.
.
**Sarc String Solvent Spherical Shell**
**The sphere center is the final image the Sarc string prior to
   exit**
**The convention here changes somewhat to ‘‘CenterIndex X Y Z’’**
**It has the form ‘‘sphere x y z Radius’’**
114 43 -6.070 11.422 2.843
115 44 -7.456 13.587 6.381 sphere -4.601 12.270 1.081 12.5
116 52 -6.968 13.153 0.878
117 53 -4.153 14.633 -1.610
118 54 -6.773 15.333 -1.894
119 56 -0.554 14.928 -0.118
.
.
.
```

Appendix D. 2D System Codes and Scripts

D.1 Code for Determining Average Position of the Particle

A C code which calculates the average distance of the particle in the system on a given milestoning boundary.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

main (int argc, char * argv[]) {


int i;
double time, time_old, cell_in, cell_to, cell_to_old, av_numer;
double time_st,sum_x,sum_y;
double average;
char ifile[255];


/*At run time define input. Need to define input file (
   time_cell**.dat) */
for (i=1;i<argc;i++) {
    if (!strcmp(argv[i],"-ifile")) {
      sscanf(argv[++i],"%s",&ifile);
    }

  }

/*Input file 1 (master cell)*/
FILE *in_file1;

char *mode = "r";
in_file1 = fopen(ifile, mode);
if (in_file1 == NULL) {
        printf("No data available for average.\n");  //If input file
            does not exist, exit program
        exit (0);
}

/*Note fscanf requires EXACT formating*/
/*Read input file and average*/
float a = 0, b = 0, c = 0, d = 0, e = 0;
int av_count=0;
int num_transitions = 1;
double co;
sum_x = 0;
sum_y=0;
```

```
cell_to=0;
while (fscanf(in_file1, "%f %f %f %f %f\n", &a, &b, &c, &d, &e) ==
   5) {

 sum_x += c;
 sum_y += d;
 cell_to += 1;
 /*printf("%.3lf %.3lf %.3lf\n",time, cell_in, cell_to);*/


 /* Averaging Block*/
 }
 printf("%.6lf %.6lf %.6lf %.6lf\n", cell_to, sum_x/cell_to, sum_y/
    cell_to,0);
 }
```

D.2 Reparameterization Code

A C code which reparameterizes the tesselation based on a series of connecting line segments.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

/*String Parameterization for 2D Toy System*/
/* Must supply number of string images and 'new' locations*/
/* Develops piecewise linear curve and deposits next*/
/* iteration on it.*/
/* compiles as gcc FILENAME.c -lm -o {Name You Like}*/


main (int argc, char * argv[]) {


int i,num_images;
double time, time_old, cell_in, cell_to, cell_to_old, av_numer;
double strim,str,time_st,sum_x,sum_y,distance,tot_len[100];
double average, x[100], y[100],f[100];
char data[255];

/*At run time define input*/
for (i=1;i<argc;i++) {
    if (!strcmp(argv[i],"-strim")) {
      sscanf(argv[++i],"%lf",&strim);
    }
    else if (!strcmp(argv[i],"-data")) {
      sscanf(argv[++i],"%s",&data);
    }
  }
```

```
str = strim;
/*Input file*/
FILE *in_file1;
char *mode = "r";
in_file1 = fopen(data, mode);
if (in_file1 == NULL) {
        printf("No data supplied for linear interpolation!\n");  //
           If input file does not exist, exit program
        exit (0);
}
if (str == 0) {
        printf("Number of string images not supplied.\n");  //If
           input file does not exist, exit program
        exit (0);
}

/*Note fscanf requires EXACT formating*/
/*Read input data file into array*/
float a = 0, b = 0, c = 0, g = 0;
num_images=0;
i=1;
while (fscanf(in_file1, "%f %f %f %f\n", &a, &b, &c, &g) == 4) {
 x[i] = b;
 y[i] = c;
 printf("Point and image # %.6lf %.6lf %i\n",x[i], y[i], num_images)
    ;
 i+=1;
 num_images+=1;
}

tot_len[0]=0;
/*Calculate distance between each pair of points. Store*/
/*Calculate total length of piecewise function*/
double d[num_images-1];
for (i=1;i<=num_images-1;i++){
        distance = ((x[i+1]-x[i])*(x[i+1]-x[i])) + ((y[i+1]-y[i])*(y
           [i+1]-y[i]));
        d[i] = pow(distance,0.5);
        tot_len[i]=tot_len[i-1]+pow(distance,0.5);
        printf("d[i] & tot_len[i] %.6lf %.6lf\n",d[i],tot_len[i]);
}

/*Determine new spacing*/
double new_spacing,x_new,y_new,curr,len;
double scaling,x_diff,y_diff;
int j,index,y_index;
new_spacing = tot_len[num_images-1]/(num_images-1); /*0.5 for test
   data*/
printf("new spacing %0.6lf\n",new_spacing);

/*Point 1 Fixed*/
/****************/
```

```
printf("%.6lf %.6lf %.6lf\n",x[1],y[1],0.000000);
/****************/

/*Counting forward so start at 1*/
/*Loop for all points*/
for (i=1;i<=num_images-2;i++){
len = tot_len[num_images-1];
/*Current Location along curve*/
curr = new_spacing * i;

/*Find segment corresponding to current location*/
 for (j=num_images;j>0;j--){
        printf("len %.6lf\n",len);
        len-=d[j];
        printf("len-new %.6lf\n",len);
        if (len >= curr){
                printf("curr on segment d[%i] %.6lf\n", j,d[j]);
                index=j;
        }
  }
/*Index gives me the end point of the*/
/*segment that is longer*/
/*then my current distance along the string*/

/*Scaling Factor*/

 scaling = ((curr - tot_len[index-2])/d[index-1]);
 if (i == 1 && index==2){
  scaling = (curr)/d[1];
  /*printf("scaling factor 1 %.6lf\n", scaling);*/
 }
 /*printf("scaling factor  %.6lf\n", scaling);*/



/*X&Y Differences*/

x_diff = (x[index]-x[index - 1]);
y_diff = (y[index]-y[index - 1]);
/*printf("differences %.6lf %.6lf\n", x_diff,y_diff);*/

/*New X and Y*/
/*printf("x_diff*scaling %.6lf %.6lf\n",
   x_diff*scaling,y_diff*scaling);*/
x_new = (x_diff * scaling) + x[index - 1];
y_new = (y_diff * scaling) + y[index - 1];

printf("%.6lf %.6lf %.6lf\n",x_new,y_new,0);
}
/*end point fixed*/
/****************/
printf("%.6lf %.6lf %.6lf\n",x[num_images],y[num_images],0);
```

```
/*****************/
}
```

D.3 Code to Determine X Minimum and Maximum

A C code which determines the maximum and minimum values of the X-coordinate of the particle. Originally used for helping update the tesselation during reparameterization.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

/*Find x_min and x_max for determining the tesselation boundary
   line*/
/* Must grep for the line you want (i.e. 4 3 or 4 5)*/
/* Code reads in the data to find the extremes*/
/* for X and the corresponding Y values*/
/* You will need to pipe the values to a file at runtime */
/* compiles as gcc find_xminmax.c -lm -o {Name You Like}*/


main (int argc, char * argv[]) {


int i,num_images;
double time, time_old, cell_in, cell_to, cell_to_old, av_numer;
double strim,str,time_st,sum_x,sum_y,distance,tot_len[100];
double average, x_min, x_max, x_minref, x_maxref, y_min, y_max;
char data[255];

/*At run time define input*/
for (i=1;i<argc;i++) {
    if (!strcmp(argv[i],"-data")) {
      sscanf(argv[++i],"%s",&data);
    }
    }

/*Input file*/
FILE *in_file1;
char *mode = "r";
in_file1 = fopen(data, mode);
if (in_file1 == NULL) {
        printf("No data file for x_min x_max!\n");  //If input file
           does not exist, exit program
        exit (0);
}


/*Note fscanf requires EXACT formating*/
```

```
/*Read input data file into array*/
float a = 0, b = 0, c = 0, g = 0, h = 0;
num_images=0;
i=1;
x_minref = 100;
x_maxref = -100;


while (fscanf(in_file1, "%f %f %f %f %f\n", &a, &b, &c, &g, &h) ==
   5) {
 x_min = c;
 x_max = c;
 y = g;

if (x_min < x_minref) {
  x_minref = x_min;
  y_min = g;
}
if (x_max > x_maxref) {
 x_maxref = x_max;
 y_max = g;
}

}

/*Output X-min,y and X-max, y
/*****************/
printf("%.6lf %.6lf %.6lf %.6lf\n",x_minref, y_min, x_maxref, y_max)
   ;
/*****************/

}
```

D.4 Simple Histogram Code

A Fortran code for histogramming the data from the simulation. This version only works with straight lines.

```
program HISTOGRAM2D


 real x_min,x_max,y_min,y_max, summ,loopvalue,factor, summ2, RMSD
 integer i,frame,j,indice,step,counter,counter2
 real probability(72,72), average(72,72)
 real counters2(72,72), prob2(72,72), upper, lower, u_in, l_in
 character(10) :: filein
 character(50) :: datafile
 real,allocatable :: binX(:), binY(:),counters(:)
 real data_incr,cellin,cellto,x,y,z
 integer :: stat
```

```
!Open the input file
!Must have form datafile, upper bound, lower bound, num_bins
!.....................................................
open (unit=3, file= 'input.dat')
read (3,*) datafile, u_in, l_in, num_bins
open (unit=4, file= datafile)
!print*, "Input Data"
!print*, "data,upper,lower,bins", datafile, u_in, l_in, num_bins

!Create bins for data
!For 2d_test data, only binning Y data.
!...................
allocate(binX(num_bins),binY(num_bins),counters(num_bins))
data_incr = (u_in - l_in)/num_bins
do i = 1,num_bins
  binX(i) = l_in + i*data_incr
  binY(i) = l_in + i*data_incr
  counters(i) = 0
  !print*, "Bin Y-value", binY(i)
end do

counter = 0
do
   !Set up for xition file format!!!!!!!
   read(4,*,iostat=stat) cellin,cellto,x,y,z
   if (stat /= 0) exit
  do j=2,num_bins
   do i=2,num_bins
      if (y .lt. binX(j) .and. y .gt. binX(j-1) .and y .lt. binY(i)
         .and. y .gt. binY(i-1)) then
        counters(i) = counters(i) + 1 !Counters(i,j) is the matrix
           containing the # points in each bin
        counter = counter + 1 !Keeps track of how many points are
           binned
        !print*, "Found bin!", i,y,binY(i),binY(i-1)
      end if
   end do

end do
!Output bin,binValue,num points in bin
!do i=1,num_bins
 !print*,i,binY(i),counters(i)
!end do

!Normalize bin values by number of points and output
do i=1,num_bins
 print*, i,binY(i),(counters(i)/counter)
 summ = summ + (counters(i)/counter)
end do
```

```
 !print*, summ


end
```

D.5 Histogram Code with Coordinate Transformation

A Fortran code for histogramming the data from the simulation. This version performs a coordinate transformation, and can be used for iterations after the first.

```
program HISTOGRAM2D


 real x_min,x_max,y_min,y_max, summ,loopvalue,factor, summ2, RMSD
 integer i,frame,j,indice,step,counter,counter2
 real probability(72,72), average(72,72)
 real counters2(72,72), prob2(72,72), upper, lower, u_in, l_in
 character(10) :: filein
 character(50) :: datafile
 real,allocatable :: binX(:), binY(:),counters(:), tess_x(:), tess_y
    (:)
 real, allocatable :: areaY(:)
 real data_incr,x,y,z, area_tot
 integer :: stat, num_bins, numpoints, cellin, cellto
 real m1, m2, angle, x_pr1, x_pr2, y_pr1, y_pr2
 real x_in, y_in, z_in, mp_rotX, mp_rotY, sl_rot, sl_bline
 real y_line, x_line, mpX

open(unit=4, file= 'input_line.dat')
read(4,*) datafile, num_bins, numpoints
open(unit=5, file= datafile)
read(5,*) cellin, cellto, x_in, y_in, z_in
open(unit=3, file= 'tesselation.dat')
!Read in and store tesselation
!.............................
num_bins = num_bins + 1
allocate(tess_x(numpoints),tess_y(numpoints))
  do i=1,numpoints
    read(3,*) x, y
    tess_x(i) = x
    tess_y(i) = y
    !print*, tess_x(i), tess_y(i)
  end do

!Determine max and min of x&y data
stat = 0
x_min = 100
x_max = -2
y_min = 100
y_max = -2
```

```
mpX = (tess_x(cellin) + tess_x(cellto))/2.0
  do while (stat .eq. 0)
    read(5,*,iostat=stat) cellin, cellto, x_in, y_in, z_in
    if (x_in .lt. x_min) then
        x_min = x_in
    end if
    if (x_in .gt. x_max) then
        x_max = x_in
    end if
      if (y_in .lt. y_min) then
        y_min = y_in
    end if
    if (y_in .gt. y_max) then
        y_max = y_in
    end if

   end do
   !print*, "Data Range X", x_min, x_max
   !print*, "Data Range Y", y_min, y_max
   close(5)
!Calculate Slopes
!................
m1 = (tess_y(numpoints) - tess_y(1))/(tess_x(numpoints) - tess_x(1))
    !Ref Slope

!do i=2, numpoints
        m2 = (tess_y(cellin) - tess_y(cellto))/(tess_x(cellin) -
           tess_x(cellto)) !Slope adjacent pts

        !Find Angle between m1 and m2
        !............................
        angle = -1*atan(m1 - m2 / (1.0 + m1*m2)) !Returns radians
        !Multiplied by -1 because want to shift to axis, not away
           from
        !print*, m2, angle

        !xform to rotated coord system
        x_pr1 = cos(angle)*tess_x(cellin) + sin(angle)*tess_y(cellin
           )
        y_pr1 = -1*sin(angle)*tess_x(cellin) + cos(angle)*tess_y(
           cellin)
        x_pr2 = cos(angle)*tess_x(cellto) + sin(angle)*tess_y(cellto
           )
        y_pr2 = -1*sin(angle)*tess_x(cellto) + cos(angle)*tess_y(
           cellto)
        !print*, "Rotated cell centers"
        !print*, x_pr1, y_pr1
        !print*, x_pr2, y_pr2
        !print*, ""

        !Slope and midpoint in rotated coord system
        sl_rot = (y_pr2 - y_pr1) / (x_pr2 - x_pr1)
```

```
        mp_rotX = (x_pr2 + x_pr1)/2.0
        mp_rotY = (y_pr2 + y_pr1)/2.0

        !Slope line to bin on
        sl_bline = -1 * (1.0/sl_rot)

        !x&y min/max Rotated (Always bin along Y, so X = constant,
           i.e. mpX)
        !This is the range of data to bin over
        x_pr1 = cos(angle)*mpX + sin(angle)*y_min
        y_pr1 = -1*sin(angle)*mpX + cos(angle)*y_min
        x_pr2 = cos(angle)*mpX + sin(angle)*y_max
        y_pr2 = -1*sin(angle)*mpX + cos(angle)*y_max
        !print*, "Rotated min/max", y_pr1, y_pr2

        !Determine bins
        allocate(binY(num_bins),counters(num_bins),areaY(num_bins))
        data_incr = abs(y_pr2 - y_pr1)/(num_bins-1)
        do i=1 ,num_bins
          binY(i) = y_pr1 + data_incr*(i-1)
          counters(i) = 0
          !print*, "Bin",i,"Value",binY(i)
        end do

        counter = 0
        !Reopen file to bin
        open(unit=5, file= datafile)
        stat = 0
do
   !Set up for xition file format!!!!!!!
   read(5,*,iostat=stat) cellin,cellto,x,y,z
   !Rotate the current point
   y_pr1 = -1*sin(angle)*mpX + cos(angle)*y
    if (stat /= 0) exit
   do i=2,num_bins
      if (y_pr1 .le. binY(i) .and. y_pr1 .ge. binY(i-1)) then
        counters(i) = counters(i) + 1 !Counters(i,j) is the matrix
           containing the # points in each bin
        counter = counter + 1 !Keeps track of how many points are
           binned
        !print*, "Found bin!", i,y,y_pr1,binY(i),binY(i-1)
      end if
   end do

end do

!Determine area each bin and total area
area_tot = 0
do i=1,num_bins-1
 areaY(i) = abs(binY(i) - binY(i+1))*counters(i)
 area_tot = area_tot + areaY(i)
 print*, area_tot
```

```
end do


!Output bin,binValue,num points in bin
!do i=1,num_bins
 !print*,i,binY(i),counters(i)
!end do

!Normalize bin values by number of points
!as well as total area and output
!Rotate back to orig coord syste
do i=1,num_bins
 y = mp_rotX*sin(angle) + binY(i)*cos(angle)
 print*, i,mpX,y,(counters(i)/counter), counters(i),(areaY(i)/
    area_tot)
 summ = summ + (counters(i)/counter)
end do
 print*, summ, "Tot Points", counter


!end do

end
```

D.6 Boltzman Weighted Distribution Code

A Fortran code for determining the Boltzman weighted distribution of the data as well as the potential itself. Useful for making sure parameters that have been changed still conform to the correct statistics.

```
!Compiles as:
!gfortran NAME.f90 -o OUTPUT -L/usr/lib64/ -llapack -lblas
module free_energy
contains
  !Will need to feed the function x,y
  function fr_en(x,y)
     implicit none
     integer, parameter                    :: dp =
        selected_real_kind(15, 307)
     real(dp), intent(in)                  :: x, y
     real(dp)                              :: fr_en, fe_x, fe_y
     real(dp)                              :: analy_fe
     real(dp)                              :: funcA, free_energy

     !Accept input
        fe_x = x
        fe_y = y
        !print*, "Input X&Y", fe_x, fe_y
```

```
     !My Free Energy Function
     funcA = 3._dp*exp(-fe_x**2-(fe_y-(1._dp/3._dp))**2) &
        - 3._dp*exp(-fe_x**2-(fe_y-(5._dp/3._dp))**2) &
        - 5._dp*exp(-(fe_x-1._dp)**2-fe_y**2) &
        - 5._dp*exp(-(fe_x+1._dp)**2-fe_y**2) + 0.2_dp*fe_x**4 &
        + 0.2_dp*(fe_y-(1._dp/3._dp))**4
        !print*, "F.E. Func", funcA
      !Free_Energy
      fr_en = funcA

 end function
end module


program BOLTZMAN_DISTRIBUTION
  use free_energy
  implicit none

  ! let us test it
  integer, parameter      :: degree = 5
  integer, parameter      :: dp = selected_real_kind(15, 307)
  integer                 :: i, j, cellnum, stat, cellin, cellto
  real(dp) :: x, y, hist_prob, k, T, prob, denom_boltz, e
  real(dp) :: degen, area, midX, num_bins, numpoints
  character(50) :: hdatafile
  real,allocatable        :: tess_x(:), tess_y(:)

  !Need to read in histogram for curve fitting
 open(unit=5, file= 'input_boltz.dat')
 read(5,*) hdatafile, cellin, cellto, numpoints
 close(5)
!open(unit=3, file= hdatafile)
!read(5,*) cellin, cellto, x_in, y_in, z_in


  open (unit=4, file= hdatafile)
  !read (3,*) cellnum, x, y, hist_prob, numpoints
  !open(unit=4, file= datafile)
  k = 1.98723310870534e-3_dp !kcal/mol
  T = 310 !Kelvin
  stat = 0
  denom_boltz = 0

  !Calculate Denominator
  do while (stat .eq. 0)
    read(4,*,iostat=stat) cellnum, x, y, hist_prob, numpoints!, area
    !print*, cellnum, x, y, hist_prob, numpoints
    !x = midX
    e = fr_en(x,y)
    !print*, "denom", denom_boltz
    denom_boltz = denom_boltz + exp((-1._dp*e)/(k*T))
```

```
  end do

  !Close data file
  !print*, "denom", denom_boltz
  !print*, "Closing and reopening data file"
  close(4)

  !Reopen file to calculate probabilities
  open(unit=4, file= hdatafile)
  stat = 0
  do while (stat .eq. 0)
    read(4,*,iostat=stat) cellnum, x, y, hist_prob, numpoints!, area
    !x = midX
    e = fr_en(x,y)
    prob =  exp((-1_dp*e)/(k*T))/denom_boltz
    print*, "x,y,probability", x, y, prob
  end do
  close(4)
end program
```

D.7 Clustering Code

A Fortran code for data clustering. Outputs the cluster with the highest probability.

```
!Cluster data to output bin values and cluster i.d to stdout
! 1.) Reads an input file of 1 line with format "datafile", "
   histogram_min", "histogram_max"
! 2.) Dynamically allocate the matrix arrays
! 3.) At run time decide bin size

program CLUSTER

 integer i,j,counter,counter_old
 character(50) :: datafile
 real col, x_val, y_val, hist_min, hist_max, bin_val
 integer :: stat, num_clusters
 real threshold, max_it, area, bin_val_old, cen_x, cen_y, cen_x2,
    cen_y2
 real midx, midy,flag

!Open the input file
!.....................................................
open (unit=3, file= 'histminmax_input.dat')
read (3,*) datafile, hist_min, hist_max, max_it
open (unit=4, file= datafile)

!Center initial values
 cen_x = 0
 cen_y = 0
```

```
!Area initially zero
area = 0
!Threshold initially zero
threshold = 0
!Cluster i.d. start
i = 1
!Start of iterations
j = 0

!Counter to ensure at least one data point exists in cluster before
   incrementing
counter = 0

!Open datafile here b/c of goto
42 open (unit=4, file= datafile)
!Threshold for iteration j = 1,2,3...
j = j + 1
threshold = hist_min + j*((hist_max - hist_min)/(max_it))
!threshold = j*hist_min
!print*, "Threshold", threshold, " 0.5", " Iteration", j
!print*, "Threshold", threshold, "-0.5", " Iteration", j
!print*, ""
!print*, "Iteration", j
!Where the magic starts
do while (threshold < hist_max)

!Read file with bin values
read(4,*,iostat=stat) col,x_val,y_val,bin_val

!If end of file
   if (stat /= 0) then
    !print*, ""
    close(4)
    !Reset cluster id and store for end of program condition
    counter_old = i
    i = 1
    if ((counter_old - 1) .eq. 1 .and. j > 1) then
     goto 3 !Exit command
    end if
    counter = 0
    bin_val_old = 0
    goto 42
   end if


!print*, "Iteration", j, "threshold", threshold
!Clustering
!.....................
if (bin_val > threshold) then
 !Assign to cluster i
 counter = counter + 1
 !Output for cluster
```

```
 !.................................
 !print*, i,x_val,y_val,bin_val,counter
 !.................................
 !Determine cluster center as largest peak
 if (bin_val > bin_val_old) then
   bin_val_old = bin_val
   cen_x = x_val
   cen_y = y_val
 end if

end if
!Assign new cluster i.d. only if previous data exists
if (bin_val < threshold .and. counter > 0) then
 i = i + 1
 !Reset counter for new cluster
 counter = 0
 !Output accumulated area
 !print*, "Cluster", i-1, "probability", area

 !output center and reset
 !print*, "Cluster", i-1, "probability", area, "Center", cen_x,
    cen_y

  !Reset bin area
 area = 0
 bin_val_old = 0
end if
!Continue to end of file

end do

!output center with greatest probability
3 print*, cen_x, cen_y

2 end
```

D.8 Script to Run and Extract Data

A bash script for running an entire iteration. Also set up to perform reparameterization as well. This is the master script to run and edit when performing tests on the 2D system.

```
#!/bin/bash

#Loop over the number of cells you have. MAKE SURE TO CHANGE THIS!
it="0"
#First data point is fixed
#echo "0 -1 0 0" >> midpoints_hist_it$it.dat
for (( c=2; c<=10; c++))
```

```
do

#Define home-cell and the line from tesselation data you need
     homecell="$c"
     line=$(sed $c!d tesselation.dat)
     #cell=$(expr $c - 1)
#For The template file copy it and rename .old (so as not to destroy
    the good template)
     for fl in ajb1_template.ini; do
     cp $fl $fl.old
#Check for patterns and replace with replacement variables. Then
   pipe it to a new file that is numbered according to the loop
   index
     sed  -e "s/hc/$homecell/g" -e "s/xy_coord/$line/g" $fl.old >
        ajb1_1500gam_lowT_cell$c.ini
     done
#Run the cell, extract the hitting data and transition data
./odh ajb1_1500gam_lowT_cell$c.ini > cell"$c"_1500gam_lowT_it$it.log
tclsh ms_extract.tcl cell"$c"_1500gam_lowT_it$it
tclsh transitions_only.tcl cell"$c"_1500gam_lowT_it$it
#Calculate the average position based on transitions only
#./av_individual -ifile xition_cell"${c}"_9cell_it$it.dat >>
   midpoints_9cell_it$it.dat


done
#Final point fixed
#echo "0 1 0 0" >> midpoints_9cell_it$it.dat
#Reparameterize and build new tesselation file.
#Note you will need to copy the tesselation_it$it.dat file to be
   tesselation.dat
#before running again.
#./param -data midpoints_9cell_it$it.dat -strim 9.0 >
   reparam_9cell_it$it.dat
#cut -d' ' -f1,2 reparam_9cell_it$it.dat > tesselation_it$it.dat
```

D.9 Input Script for a Single Particle System

Sample input script for the 2D system.

```
!------Input Card for ODH program

100000    ! steps  (x100)
1       ! write steps (x100)
100      ! output chapeau steps  (x100)
75.0         ! T
5.0           ! gama
 .false.      ! TAMD?
 .false.      ! Measure escape rate?
750           ! auxiliary T
```

```
10000.0        ! gamaz
1500.0         ! spring constant
 .true.        ! milestoning
5              ! milestoning home cell
-0.2 0.0           !xini
```

Appendix E. Topology and Parameter Files for FOA and FADH

E.1 FADH Parameter File

CHARMM style parameters for the cofactor FADH.

```
! quick-and-dirty CHARMM parameterization of
! reduced (anion) flavin adenine dinucleotide FAD
! (c) 2010 cfa
!
! PARAMETERS ARE NOT OPTIMIZED
!
! FAD is based on the parameterization of NAD:
! - The adenine and phosphate parameters are unchanged; some
!   atom names are changed to keep consistent with PDB format
!   for FAD.
! - Sugar atoms on the flavin side of the phosphates are borrowed
!   from NAD, even though the flavin sugar is not cyclic!
! - all flavin atoms are given new atom types EXCEPT the methyl
!   and aromatic carbons and hydrogens that are not bound
!   directly to any nitrogen.
! - all other atom types, bond/angle/dihedral parameters are
!   guessed from existing analogs.  These guesses involve
!   ONLY the flavin ring.
!
! This file depends on existing parameters in
! par_all22_prot_cmap.prm and par_all27_na.prm

BONDS
! new bonds for acyclic sugar
CN7B  CN8B   200.0       1.450
CN8B  HN7    309.0       1.111  !Alkanes, sacred

! connection from sugar to flavin
CN8B  FN10   220.0       1.520

FN10  FC10   302.0       1.392
FN10  FC9A   302.0       1.420

FC10  FN1    400.0       1.330
FN1   FC2    260.0       1.370
FC2   FO2    620.0       1.210
FC2   FN3    260.0       1.390
FN3   FH3    440.0       1.000
FN5   FH5    440.0       1.000
FN3   FC4    260.0       1.420
FC4   FO4    620.0       1.240
FC4   FC4X   250.0       1.400
FC4X  FC10   250.0       1.400
```

```
FC4X  FN5     400.0       1.390
FN5   FC5X    310.0       1.370
FC5X  FC9A    305.0       1.420
FC5X  FC6     305.0       1.420
FC6   FH6     340.0       1.000
FC6   FC7     305.0       1.380
FC7   CT3     230.0       1.510
FC7   FC8     305.0       1.450
FC8   CT3     230.0       1.440
FC8   FC9     305.0       1.420
FC9   FH9     340.0       1.000
FC9   FC9A    305.0       1.420

! connection to cys SG from C8M
FC8   CT2     230.0       1.510

ANGLES

CN7B CN8B HN7      33.4      110.10  22.53   2.179
CN8B CN7B HN7      33.4      110.10  22.53   2.179
CN8B CN7B CN7     110.0       96.0
ON5  CN7B CN8B     80.0      108.4  !
HN7  CN8B  HN7     35.5      109.00   5.40   1.802

! connection to flavin
CN7B CN8B FN10    110.0      111.0
HN7  CN8B FN10     43.0      111.0
CN8B FN10 FC9A     45.0      120.0
CN8B FN10 FC10     45.0      120.0

! flavin internal to ring 1
FC10 FN1  FC2      90.0      120.0
FN1  FC2  FO2      50.0      122.5
FO2  FC2  FN3      80.0      120.0
FN1  FC2  FN3      60.0      116.0
FC2  FN3  FH3      34.0      116.2
FC2  FN3  FC4      50.0      125.5
FH3  FN3  FC4      34.0      118.3
FN3  FC4  FO4      80.0      120.0
FN3  FC4  FC4X     80.0      115.4
FO4  FC4  FC4X     80.0      125.0
FC4  FC4X FC10     52.0      117.5
FC4X FC10 FN1      60.0      125.6

! ring 1 to ring 2
FC4  FC4X FN5      60.0      115.0
FN1  FC10 FN10     60.0      118.5

! ring 2
FC4X FN5  FH5      34.0      120.0
FC5X FN5  FH5      34.0      120.0
FC4X FN5  FC5X     60.0      114.5
```

```
FN5   FC5X FC9A     60.0      122.8
FC5X  FC9A FN10     60.0      119.0
FC9A  FN10 FC10     60.0      120.0
FN10  FC10 FC4X     60.0      116.0
FC10  FC4X FN5      60.0      127.4

! ring 2 to ring 3
FN5   FC5X FC6      60.0      116.8
FC9   FC9A FN10     60.0      121.7

! ring 3
FC5X FC6   FC7      40.000     120.00    35.00    2.41620
FC5X FC6   FH6      30.000     120.00    22.00    2.15250
FH6  FC6   FC7      30.000     120.00    22.00    2.15250
FC6  FC7   CT3      45.000     120.00
FC6  FC7   FC8      40.000     120.00    35.00    2.41620
FC7  CT3   HA       29.000     120.00
CT3  FC7   FC8      45.000     120.00
FC7  FC8   CT3      45.000     120.00
FC8  CT3   HA       29.000     120.00
FC7  FC8   FC9      40.000     120.00    35.00    2.41620
CT3  FC8   FC9      45.000     120.00
FC8  FC9   FH9      30.000     120.00    22.00    2.15250
FC8  FC9   FC9A     40.000     120.00    35.00    2.41620
FH9  FC9   FC9A     30.000     120.00    22.00    2.15250
FC9  FC9A  FC5X     40.000     120.00    35.00    2.41620
FC9A FC5X  FC6      40.000     120.00    35.00    2.41620

! due to connection to cys 315
CT2  FC8   FC9      45.000     120.00
FC8  CT2   HA       29.000     120.00
CT2  FC8   FC7      45.000     120.00
FC8  CT2   SM       58.000     112.50
CT2  SM    FC8      34.000     95.0000

DIHEDRALS

ON5  CN7   CN8B ON2        3.4      1        180.0
CN8B CN7   ON5  HN5        0.5      3          0.0
CN8B CN7B  CN7  CN7        0.4      6          0.0
ON5  CN7   CN7  CN7B       0.8      6          0.0 !
ON5  CN7   CN7  CN7B       0.4      5          0.0 ! Moves the barrier
   right
ON5  CN7   CN7  CN7B       2.0      3        180.0 !
CN7  CN7B  CN8B NN2        0.0      3          0.0
HN7  CN8B  CN7B CN7        0.195    3          0.0
HN7  CN8B  CN7B HN7        0.000    3          0.0
CN8B CN7B  CN7  HN7        0.195    3          0.0
ON5  CN7   CN7B CN8B       0.8      6          0.0 !
ON5  CN7B  CN8B HN7        0.0      3          0.0 !
HN5  ON5   CN7B CN8B       0.000    6        180.0 !
HN5  ON5   CN7B CN8B       0.000    3          0.0 !
```

```
! connection
CN7   CN7B  CN8B  FN10       0.3      6        180.0
HN7   CN7B  CN8B  FN10       0.3      6        180.0
ON5   CN7B  CN8B  FN10       0.3      6        180.0
CN7B  CN8B  FN10  FC10       0.3      6          0.0
HN7   CN8B  FN10  FC10       0.3      6          0.0
CN7B  CN8B  FN10  FC9A       0.3      6          0.0
HN7   CN8B  FN10  FC9A       0.3      6          0.0
CN8B  FN10  FC10  FN1        2.0      2        180.0
CN8B  FN10  FC10  FC4X       2.0      2        180.0
CN8B  FN10  FC9A  FC9        2.0      2        180.0
CN8B  FN10  FC9A  FC5X       2.0      2        180.0

! flavin ring 1
FN1   FC2   FN3   FC4        3.0      2        180.0
FN1   FC2   FN3   FH3        3.5      2        180.0
FO2   FC2   FN3   FC4       14.0      2        180.0
FO2   FC2   FN3   FH3        3.5      2        180.0
FC2   FN3   FC4   FC4X       3.5      2        180.0
FC2   FN3   FC4   FO4       14.0      2        180.0
FH3   FN3   FC4   FC4X       3.5      2        180.0
FH3   FN3   FC4   FO4        3.5      2        180.0
FN3   FC4   FC4X  FC10       3.0      2        180.0
FO4   FC4   FC4X  FC10      14.0      2        180.0
FC4   FC4X  FC10  FN1        3.0      2        180.0
FC4X  FC10  FN1   FC2        3.0      2        180.0
FC10  FN1   FC2   FN3        3.0      2        180.0
FC10  FN1   FC2   FO2       14.0      2        180.0

! ring 1 to ring 2
FN10  FC10  FN1   FC2        3.0      2        180.0
FN10  FC10  FC4X  FC4        3.0      2        180.0
FN1   FC10  FN10  FC9A       3.0      2        180.0
FN1   FC10  FC4X  FN5        3.0      2        180.0
FC4   FC4X  FC10  FN10       3.0      2        180.0
FO4   FC4   FC4X  FN5       14.0      2        180.0
FN3   FC4   FC4X  FN5        3.0      2        180.0
FC4   FC4X  FN5   FC5X       3.0      2        180.0

! ring 2
FC4X  FN5   FC5X  FC9A       3.0      2        180.0
FN5   FC5X  FC9A  FN10       3.0      2        180.0
FC5X  FC9A  FN10  FC10       3.0      2        180.0
FC9A  FN10  FC10  FC4X       3.0      2        180.0
FN10  FC10  FC4X  FN5        3.0      2        180.0
FC10  FC4X  FN5   FC5X       3.0      2        180.0
FC9A  FC5X  FN5   FH5        3.5      2        180.0
FC4   FC4X  FN5   FH5        3.5      2        180.0
FC10  FC4X  FN5   FH5        3.5      2        180.0
FH5   FN5   FC5X  FC6        3.5      2        180.0
```

```
! ring 2 to ring 3
FC10 FN10 FC9A FC9        3.0      2        180.0
FN10 FC9A FC9  FH9        3.5      2        180.0
FN10 FC9A FC9  FC8        3.0      2        180.0
FN10 FC9A FC5X FC6        3.0      2        180.0
FC4X FN5  FC5X FC6        3.0      2        180.0
FN5  FC5X FC9A FC9        3.0      2        180.0
FN5  FC5X FC6  FH6        3.0      2        180.0
FN5  FC5X FC6  FC7        3.0      2        180.0

! ring 3
FC5X FC6  FC7  FC8        3.0      2        180.0
FC5X FC6  FC7  CT3        3.0      2        180.0
FH6  FC6  FC7  FC8        3.0      2        180.0
FH6  FC6  FC7  CT3        3.0      2        180.0
FC6  FC7  CT3  HA         0.0      2        180.0
FC8  FC7  CT3  HA         0.0      2        180.0
FC6  FC7  FC8  CT3        3.0      2        180.0
FC6  FC7  FC8  FC9        3.0      2        180.0
FC7  FC8  CT3  HA         0.0      2        180.0
FC9  FC8  CT3  HA         0.0      2        180.0
FC7  FC8  FC9  FH9        3.5      2        180.0
FC7  FC8  FC9  FC9A       3.0      2        180.0
CT3  FC7  FC8  CT3        3.0      2        180.0
CT3  FC7  FC8  FC9        3.0      2        180.0
CT3  FC8  FC9  FH9        3.0      2        180.0
CT3  FC8  FC9  FC9A       3.0      2        180.0
FC8  FC9  FC9A FC5X       3.0      2        180.0
FH9  FC9  FC9A FC5X       3.5      2        180.0
FC9  FC9A FC5X FC6        3.0      2        180.0
FC9A FC5X FC6  FC7        3.0      2        180.0
FC9A FC5X FC6  FH6        3.0      2        180.0

! due to connection to cys315
CT2  SM   CT2  CT1        0.2400   1       180.00
CT2  SM   CT2  CT1        0.3700   3         0.00
CT2  SM   CT2  HA         0.2800   3         0.00
CT2  FC8  FC7  FC6        3.0      2        180.0
CT2  FC8  FC9  FH9        3.0      2        180.0
FC9A FC9  FC8  CT2        3.0      2        180.0
FC7  FC8  CT2  SM         0.0      2        180.0
FC7  FC8  CT2  HA         0.0      2        180.0
CT3  FC7  FC8  CT2        3.0      2        180.0
FC8  CT2  SM   CT2        0.2400   1       180.00
FC8  CT2  SM   CT2        0.3700   3         0.00
HA   CT2  FC8  FC9        0.0      2        180.0
FC9  FC8  CT2  SM         0.0      2        180.0

IMPROPER

NONBONDED
```

```
FH3     0.000000  -0.046000     0.224500
FH5     0.000000  -0.046000     0.224500
FH6     0.000000  -0.022000     1.320000
FH9     0.000000  -0.022000     1.320000

FC2     0.000000  -0.110000     2.000000
FC4     0.000000  -0.110000     2.000000
FC4X    0.0       -0.075        1.9000
FC5X    0.0       -0.075        1.9000
FC6     0.000000  -0.070000     1.992400
FC7     0.000000  -0.070000     1.992400
FC8     0.000000  -0.070000     1.992400
FC9     0.000000  -0.070000     1.992400
FC9A    0.0       -0.075        1.9000
FC10    0.0       -0.075        1.9000

FN1     0.0       -0.20         1.85
FN3     0.000000  -0.200000     1.850000   0.000000  -0.200000     1
    .550000
FN5     0.000000  -0.200000     1.850000   0.000000  -0.200000     1
    .550000
FN10    0.0       -0.20         1.85

FO2     0.000000  -0.120000     1.700000   0.000000  -0.120000     1
    .400000
FO4     0.000000  -0.120000     1.700000   0.000000  -0.120000     1
    .400000
```

E.2 FOA Parameter File

CHARMM style parameters for the competitive inhibitor FOA.

```
! quick-and-dirty parameterization of 2-furoate anion
BONDS
FOAC1 FOAC2    200.0     1.530
FOAC2 FOAC3    350.0     1.385
FOAC3 FOAC4    350.0     1.447
FOAC4 FOAC5    350.0     1.385
FOAC5 FOAO8    360.0     1.412
FOAO8 FOAC2    360.0     1.431
FOAO6 FOAC1    525.0     1.290
FOAO7 FOAC1    525.0     1.290
FOAC3 FOAH3    340.0     1.080
FOAC4 FOAH4    340.0     1.080
FOAC5 FOAH5    340.0     1.080

ANGLES
FOAO7 FOAC1 FOAO6   100.000    124.00   70.00   2.22500
FOAC2 FOAC1 FOAO6    65.000    118.00
FOAO7 FOAC1 FOAC2    65.000    118.00
```

```
FOAC1 FOAC2 FOAC3    45.800    131.60
FOAC1 FOAC2 FOAO8    45.800    120.30
FOAO8 FOAC2 FOAC3    45.800    108.10
FOAC2 FOAC3 FOAC4   120.000    108.30
FOAC2 FOAC3 FOAH3    32.000    124.00
FOAH3 FOAC3 FOAC4    32.000    127.70
FOAC3 FOAC4 FOAC5   120.000    107.00
FOAC3 FOAC4 FOAH4    32.000    127.30
FOAH4 FOAC4 FOAC5    32.000    125.70
FOAC4 FOAC5 FOAH5    32.000    134.60
FOAC4 FOAC5 FOAO8    45.800    109.80
FOAH5 FOAC5 FOAO8    32.000    115.60
FOAC5 FOAO8 FOAC2    95.000    106.90

DIHEDRALS
FOAO6 FOAC1 FOAC2 FOAO8      0.4000  2   180.00
FOAO6 FOAC1 FOAC2 FOAC3      0.4000  2   180.00
FOAO7 FOAC1 FOAC2 FOAO8      0.4000  2   180.00
FOAO7 FOAC1 FOAC2 FOAC3      0.4000  2   180.00
FOAC1 FOAC2 FOAC3 FOAH3      4.2000  2   180.00
FOAC1 FOAC2 FOAC3 FOAC4      3.1000  2   180.00
FOAC1 FOAC2 FOAO8 FOAC5      3.1000  2   180.00
FOAC2 FOAC3 FOAC4 FOAH4      4.2000  2   180.00
FOAC2 FOAC3 FOAC4 FOAC5      3.1000  2   180.00
FOAH3 FOAC3 FOAC4 FOAH4      2.4000  2   180.00
FOAH3 FOAC3 FOAC4 FOAC5      4.2000  2   180.00
FOAC3 FOAC4 FOAC5 FOAH5      4.2000  2   180.00
FOAC3 FOAC4 FOAC5 FOAO8      3.1000  2   180.00
FOAH4 FOAC4 FOAC5 FOAH5      2.4000  2   180.00
FOAH4 FOAC4 FOAC5 FOAO8      4.2000  2   180.00
FOAC4 FOAC5 FOAO8 FOAC2      3.1000  2   180.00
FOAH5 FOAC5 FOAO8 FOAC2      4.2000  2   180.00
FOAO8 FOAC2 FOAC3 FOAH3      4.2000  2   180.00
FOAO8 FOAC2 FOAC3 FOAC4      3.1000  2   180.00
FOAC3 FOAC2 FOAO8 FOAC5      3.1000  2   180.00

! in progress... 6/10/10
NONBONDED
FOAH3    0.000000  -0.030000     1.358200   0.000000  -0.030000
   1.358200
FOAH4    0.000000  -0.030000     1.358200   0.000000  -0.030000
   1.358200
FOAH5    0.000000  -0.030000     1.358200   0.000000  -0.030000
   1.358200
FOAC1    0.000000  -0.070000     2.000000
FOAC2    0.000000  -0.070000     1.992400
FOAC3    0.000000  -0.070000     1.992400
FOAC4    0.000000  -0.070000     1.992400
FOAC5    0.000000  -0.070000     1.992400
FOAO6    0.000000  -0.120000     1.700000
FOAO7    0.000000  -0.120000     1.700000
FOAO8    0.0       -0.1000       1.6500
```

E.3 FADH Topology File

Topology file for FADH needed to generate the MSOX psf file while building the systems to model.

```
! quick-and-dirty CHARMM topology description of
! flavin adenine dinucleotide (oxidized, resname FAD)
! (c) 2009 cfa
!
! PARAMETERS ARE NOT OPTIMIZED
!
! FAD is based on the parameterization of NAD:
! - The adenine and phosphate parameters are unchanged; some
!   atom names are changed to keep consistent with PDB format
!   for FAD.
! - Sugar atoms on the flavin side of the phosphates are borrowed
!   from NAD, even though the flavin sugar is not cyclic!
! - all flavin atoms are given new atom types EXCEPT the methyl
!   and aromatic carbons and hydrogens that are not bound
!   directly to any nitrogen.
! - all other atom types, bond/angle/dihedral parameters are
!   guessed from existing analogs.  These guesses involve
!   ONLY the flavin ring.
!
! Suggestion -- parameterize and optimize a flavin diphosphate
!
! must read after nucleic acid topologies
! this molecule is based on ADP

MASS     1 FH3     1.00800 H
MASS     2 FH6     1.00800 H
MASS     3 FH9     1.00800 H
MASS     4 FH5     1.00800 H

MASS     5 FC2    12.01100 C
MASS     6 FC4    12.01100 C
MASS     7 FC4X   12.01100 C
MASS     8 FC5X   12.01100 C
MASS     9 FC6    12.01100 C
MASS    10 FC7    12.01100 C
MASS    11 FC8    12.01100 C
MASS    12 FC9    12.01100 C
MASS    13 FC9A   12.01100 C
MASS    14 FC10   12.01100 C

MASS    15 FN1    14.00700 N
MASS    16 FN3    14.00700 N
MASS    17 FN5    14.00700 N
MASS    18 FN10   14.00700 N

MASS    19 FO2    15.99900 O
```

```
MASS    20 FO4    15.99900 O


RESI FADH        -3.00  ! reduced flavin adenine dinucleotide anion,
    cfa
GROUP                   !
ATOM C4B  CN7     0.16  !                         H6A1  H6A2
ATOM H4B  HN7     0.09  !                           \  /
ATOM O4B  ON6B   -0.50  !                            N6A
ATOM C1B  CN7B    0.16  !                            |
ATOM H1B  HN7     0.09  !                           C6A
GROUP                   !                          //  \
ATOM C5A  CN5     0.28  !                          N1A  C5A--N7A\\
ATOM N7A  NN4    -0.71  !                          |    ||
   C8A-H8A
ATOM C8A  CN4     0.34  !                          C2A  C4A--N9A/
ATOM H8A  HN3     0.12  !                         / \\ /     \
ATOM N9A  NN2    -0.05  !                      H2A  N3A      \
                        !                                     \
ATOM N1A  NN3A   -0.74  !                                      \
ATOM C2A  CN4     0.50  !                                       \
ATOM H2A  HN3     0.13  !           O2P    O1A    H5B H4B  O4B  \
ATOM N3A  NN3A   -0.75  !           |      |      |    \ /   \  \
ATOM C4A  CN5     0.43  !      O5'--P-O3P--PA-O5B-C5B---C4B    C1B
ATOM C6A  CN2     0.46  !      |    |       |      |     \     / \
                        !      |    O1P    O2A    H5B'   C3B--C2B
                           H1B
ATOM N6A  NN1    -0.77  !      |                          / \   / \
ATOM H6A1 HN1     0.38  !  H5'-C5'-H5''                O3B H3B O2B
   H2B'
ATOM H6A2 HN1     0.38  !      |                        |       |
GROUP                   !  H4'-C4'-O4'-H4''           H3T     H2B
ATOM C2B  CN7B    0.14  !        \
ATOM H2B' HN7     0.09  !H3''-O3'-C3'-H3'
ATOM O2B  ON5    -0.66  !        /
ATOM H2B  HN5     0.43  !  H2'-C2'-O2'-H2''
GROUP                   !      |
ATOM C3B  CN7     0.14  !  H1'-C1'-H1''
ATOM H3B  HN7     0.09  !       \
ATOM O3B  ON5    -0.66  !        \       H9      H8M1
ATOM H3T  HN5     0.43  !         \      |      /
GROUP                   !     N1  N10   C9   C8M-H8M2
ATOM C5B  CN8B   -0.08  !    / \\ /  \ // \ /  \
ATOM H5B  HN8     0.09  !O2=C2  C10  C9A   C8   H8M3
ATOM H5B' HN8     0.09  !   |    |    |    ||
ATOM O5B  ON2    -0.62  !H3-N3  C4X  C5X   C7   H7M3
ATOM PA   P       1.50  !    \ /  \\ / \\ / \  /
ATOM O1A  ON3    -0.82  !     C4   N5   C6   C7M-H7M2
ATOM O2A  ON3    -0.82  !     ||        |      \
ATOM O3P  ON2    -0.68  !     O4        H6      H7M1
ATOM P    P       1.50
ATOM O5'  ON2    -0.62
```

```
ATOM O1P  ON3    -0.82  ! charges for this group taken from NAD
ATOM O2P  ON3    -0.82  ! adds up to -2 (cfa; 3/27/09)
ATOM C5'  CN8B   -0.08
ATOM H5'  HN8     0.09
ATOM H5'' HN8     0.09
GROUP
ATOM C4'  CN7     0.14  ! charges for this group (and the next two)
ATOM H4'  HN7     0.09  ! taken from corresponding groups in the
   cyclic
ATOM O4'  ON5    -0.66  ! sugar
ATOM H4'' HN5     0.43
GROUP
ATOM C3'  CN7     0.14
ATOM H3'  HN7     0.09
ATOM O3'  ON5    -0.66
ATOM H3'' HN5     0.43
GROUP
ATOM C2'  CN7B    0.14
ATOM H2'  HN7     0.09
ATOM O2'  ON5    -0.66
ATOM H2'' HN5     0.43
GROUP
ATOM C1'  CN8B   -0.18 ! standard aliphatic CT2 CHARMM charges
ATOM H1'  HN7     0.09
ATOM H1'' HN7     0.09
GROUP
ATOM N10  FN10   -0.400 ! scaled partial charges loosely based on a
ATOM C9A  FC9A    0.420 ! 6-311G* optimization in Gaussian (cfa)
ATOM C10  FC10    0.640 ! of just the N10-methylated flavin
ATOM C4X  FC4X    0.000 ! (really, a lot if it is guesswork)
ATOM C5X  FC5X    0.000
ATOM N5   FN5    -0.700
ATOM H5   FH5     0.380
ATOM N1   FN1    -0.900
ATOM C2   FC2     0.630
ATOM O2   FO2    -0.700
ATOM N3   FN3    -0.700
ATOM H3   FH3     0.400
ATOM C4   FC4     0.630
ATOM O4   FO4    -0.700
GROUP
ATOM C7   FC7     0.00
GROUP
ATOM C7M  CT3    -0.27
ATOM H7M1 HA      0.09
ATOM H7M2 HA      0.09
ATOM H7M3 HA      0.09
GROUP
ATOM C8   FC8     0.00
GROUP
ATOM C8M  CT3    -0.27
ATOM H8M1 HA      0.09
```

```
ATOM H8M2 HA       0.09
ATOM H8M3 HA       0.09
GROUP
ATOM C6   FC6     -0.115
ATOM H6   FH6      0.115
GROUP
ATOM C9   FC9     -0.115
ATOM H9   FH9      0.115

BOND  O1A   PA   O2A   PA   O5B   PA   O3P   PA
BOND  C5B  O5B   C4B  C5B   H5B  C5B  H5B'  C5B
BOND  O4B  C4B   C3B  C4B   H4B  C4B   C1B  O4B
BOND  O3B  C3B   C2B  C3B   H3B  C3B   H3T  O3B
BOND  O2B  C2B   C1B  C2B  H2B'  C2B   H2B  O2B
BOND  N9A  C1B   H1B  C1B   C8A  N9A   C4A  N9A
BOND  N7A  C8A   H8A  C8A   C5A  N7A   C6A  C5A
BOND  C4A  C5A   N6A  C6A   N1A  C6A  H6A2  N6A
BOND H6A1  N6A   C2A  N1A   N3A  C2A   H2A  C2A
BOND  C4A  N3A    C2   N1   C10   N1    O2   C2
BOND   N3   C2    C4   N3    H3   N3    O4   C4
BOND  C4X   C4    N5  C4X   C10  C4X   C5X   N5
BOND   C6  C5X   C9A  C5X    C7   C6    H6   C6
BOND  C7M   C7    C8   C7  H7M1  C7M  H7M2  C7M
BOND H7M3  C7M   C8M   C8    C9   C8  H8M1  C8M
BOND H8M2  C8M  H8M3  C8M   C9A   C9    H9   C9
BOND  N10  C9A   C10  N10   C1'  N10   C2'  C1'
BOND H1''  C1'   H1'  C1'   O2'  C2'   C3'  C2'
BOND  H2'  C2'  H2''  O2'   O3'  C3'   C4'  C3'
BOND  H3'  C3'  H3''  O3'   O4'  C4'   C5'  C4'
BOND  H4'  C4'  H4''  O4'   O5'  C5'   H5'  C5'
BOND H5''  C5'     P  O5'   O1P    P   O2P    P
BOND  O3P    P    N5   H5
IMPR N6A   C6A   H6A1  H6A2       C6A   N1A   C5A   N6A
! IC's generated using paramtool cfa
IC C5B  O5B  PA   O3P   1.460  123.93   66.23  104.30  1.646
IC C5B  O5B  PA   O2A   1.460  123.93  -42.54  109.70  1.503
IC C5B  O5B  PA   O1A   1.460  123.93 -177.63  108.18  1.455
IC P    O3P  PA   O2A   1.661  126.47 -174.46  102.14  1.503
IC P    O3P  PA   O5B   1.661  126.47   71.30  104.30  1.546
IC P    O3P  PA   O1A   1.661  126.47  -44.14  109.18  1.455
IC H5B' C5B  O5B  PA    1.000  110.04   17.48  123.93  1.546
IC H5B  C5B  O5B  PA    1.000  110.06 -102.65  123.93  1.546
IC C4B  C5B  O5B  PA    1.468  107.24  137.40  123.93  1.546
IC H4B  C4B  C5B  H5B'  0.999  102.10  170.67  110.25  1.000
IC C3B  C4B  C5B  H5B'  1.542  116.50   50.87  110.25  1.000
IC O4B  C4B  C5B  H5B'  1.433  110.53  -72.41  110.25  1.000
IC H4B  C4B  C5B  O5B   0.999  102.10   50.88  107.24  1.460
IC C3B  C4B  C5B  O5B   1.542  116.50  -68.92  107.24  1.460
IC O4B  C4B  C5B  O5B   1.433  110.53  167.80  107.24  1.460
IC H4B  C4B  C5B  H5B   0.999  102.10  -68.95  110.26  1.000
IC C3B  C4B  C5B  H5B   1.542  116.50  171.26  110.26  1.000
IC O4B  C4B  C5B  H5B   1.433  110.53   47.97  110.26  1.000
```

```
IC C1B  O4B  C4B  H4B   1.384  109.90 -106.90  109.94  0.999
IC C1B  O4B  C4B  C5B   1.384  109.90  141.14  110.53  1.468
IC C1B  O4B  C4B  C3B   1.384  109.90   12.88  107.67  1.542
IC O3B  C3B  C4B  H4B   1.427  114.30    6.03  109.96  0.999
IC C2B  C3B  C4B  H4B   1.515  103.99  126.88  109.96  0.999
IC H3B  C3B  C4B  H4B   1.000  113.15 -107.37  109.96  0.999
IC O3B  C3B  C4B  C5B   1.427  114.30  121.51  116.50  1.468
IC C2B  C3B  C4B  C5B   1.515  103.99 -117.65  116.50  1.468
IC H3B  C3B  C4B  C5B   1.000  113.15    8.10  116.50  1.468
IC O3B  C3B  C4B  O4B   1.427  114.30 -113.74  107.67  1.433
IC C2B  C3B  C4B  O4B   1.515  103.99    7.11  107.67  1.433
IC H3B  C3B  C4B  O4B   1.000  113.15  132.86  107.67  1.433
IC N9A  C1B  O4B  C4B   1.423  114.95 -161.36  109.90  1.433
IC H1B  C1B  O4B  C4B   1.000  107.42   95.15  109.90  1.433
IC C2B  C1B  O4B  C4B   1.550  106.44  -27.27  109.90  1.433
IC H3T  O3B  C3B  C2B   1.001  109.00 -117.02  110.73  1.515
IC H3T  O3B  C3B  H3B   1.001  109.00  121.09   99.85  1.000
IC H3T  O3B  C3B  C4B   1.001  109.00    0.01  114.30  1.542
IC H2B' C2B  C3B  O3B   1.000  112.01 -132.39  110.73  1.427
IC C1B  C2B  C3B  O3B   1.550  102.96  101.45  110.73  1.427
IC O2B  C2B  C3B  O3B   1.432  118.08  -27.19  110.73  1.427
IC H2B' C2B  C3B  H3B   1.000  112.01  -20.02  115.23  1.000
IC C1B  C2B  C3B  H3B   1.550  102.96 -146.18  115.23  1.000
IC O2B  C2B  C3B  H3B   1.432  118.08   85.18  115.23  1.000
IC H2B' C2B  C3B  C4B   1.000  112.01  104.40  103.99  1.542
IC C1B  C2B  C3B  C4B   1.550  102.96  -21.76  103.99  1.542
IC O2B  C2B  C3B  C4B   1.432  118.08 -150.41  103.99  1.542
IC H2B  O2B  C2B  C3B   1.000  108.99    0.02  118.08  1.515
IC H2B  O2B  C2B  H2B'  1.000  108.99  116.46   92.18  1.000
IC H2B  O2B  C2B  C1B   1.000  108.99 -122.45  115.56  1.550
IC N9A  C1B  C2B  C3B   1.423  119.05  162.44  102.96  1.515
IC O4B  C1B  C2B  C3B   1.384  106.44   30.60  102.96  1.515
IC H1B  C1B  C2B  C3B   1.000  113.95  -87.61  102.96  1.515
IC N9A  C1B  C2B  H2B'  1.423  119.05   39.34  116.68  1.000
IC O4B  C1B  C2B  H2B'  1.384  106.44  -92.50  116.68  1.000
IC H1B  C1B  C2B  H2B'  1.000  113.95  149.30  116.68  1.000
IC N9A  C1B  C2B  O2B   1.423  119.05  -67.37  115.56  1.432
IC O4B  C1B  C2B  O2B   1.384  106.44  160.79  115.56  1.432
IC H1B  C1B  C2B  O2B   1.000  113.95   42.59  115.56  1.432
IC C8A  N9A  C1B  O4B   1.357  124.76   70.18  114.95  1.384
IC C4A  N9A  C1B  O4B   1.343  130.02 -105.27  114.95  1.384
IC C8A  N9A  C1B  H1B   1.357  124.76 -178.32   94.33  1.000
IC C4A  N9A  C1B  H1B   1.343  130.02    6.22   94.33  1.000
IC C8A  N9A  C1B  C2B   1.357  124.76  -57.81  119.05  1.550
IC C4A  N9A  C1B  C2B   1.343  130.02  126.74  119.05  1.550
IC N7A  C8A  N9A  C1B   1.298  113.29 -176.04  124.76  1.423
IC H8A  C8A  N9A  C1B   1.000  125.33    3.97  124.76  1.423
IC N7A  C8A  N9A  C4A   1.298  113.29    0.35  105.10  1.343
IC H8A  C8A  N9A  C4A   1.000  125.33 -179.63  105.10  1.343
IC C5A  C4A  N9A  C1B   1.354  107.50  175.78  130.02  1.423
IC N3A  C4A  N9A  C1B   1.347  125.85   -4.23  130.02  1.423
IC C5A  C4A  N9A  C8A   1.354  107.50   -0.35  105.10  1.357
```

```
IC N3A   C4A   N9A   C8A    1.347  125.85  179.64  105.10  1.357
IC C5A   N7A   C8A   N9A    1.364  104.42   -0.20  113.29  1.357
IC C5A   N7A   C8A   H8A    1.364  104.42  179.79  121.38  1.000
IC C6A   C5A   N7A   C8A    1.404  133.44 -179.63  104.42  1.298
IC C4A   C5A   N7A   C8A    1.354  109.70   -0.03  104.42  1.298
IC N6A   C6A   C5A   N7A    1.356  122.39   -0.46  133.44  1.364
IC N1A   C6A   C5A   N7A    1.351  119.51  179.38  133.44  1.364
IC N6A   C6A   C5A   C4A    1.356  122.39  179.97  116.86  1.354
IC N1A   C6A   C5A   C4A    1.351  119.51   -0.19  116.86  1.354
IC N9A   C4A   C5A   N7A    1.343  107.50    0.25  109.70  1.364
IC N3A   C4A   C5A   N7A    1.347  126.65 -179.74  109.70  1.364
IC N9A   C4A   C5A   C6A    1.343  107.50  179.92  116.86  1.404
IC N3A   C4A   C5A   C6A    1.347  126.65   -0.07  116.86  1.404
IC H6A2  N6A   C6A   C5A    1.000  119.99    0.00  122.39  1.404
IC H6A1  N6A   C6A   C5A    1.000  120.00  179.99  122.39  1.404
IC H6A2  N6A   C6A   N1A    1.000  119.99 -179.84  118.10  1.351
IC H6A1  N6A   C6A   N1A    1.000  120.00    0.15  118.10  1.351
IC C2A   N1A   C6A   C5A    1.350  118.82    0.19  119.51  1.404
IC C2A   N1A   C6A   N6A    1.350  118.82 -179.96  118.10  1.356
IC N3A   C2A   N1A   C6A    1.375  125.39    0.07  118.82  1.351
IC H2A   C2A   N1A   C6A    1.000  116.29 -179.92  118.82  1.351
IC C4A   N3A   C2A   N1A    1.347  112.77   -0.30  125.39  1.350
IC C4A   N3A   C2A   H2A    1.347  112.77  179.68  118.32  1.000
IC N9A   C4A   N3A   C2A    1.343  125.85 -179.69  112.77  1.375
IC C5A   C4A   N3A   C2A    1.354  126.65    0.30  112.77  1.375
IC N3    C2    N1    C10    1.392  116.49   -1.08  119.67  1.331
IC O2    C2    N1    C10    1.215  122.46 -179.64  119.67  1.331
IC N10   C10   N1    C2     1.391  118.44  175.09  119.67  1.369
IC C4X   C10   N1    C2     1.401  125.60   -0.86  119.67  1.369
IC H3    N3    C2    N1     1.000  116.22 -178.26  116.49  1.369
IC C4    N3    C2    N1     1.419  125.50    1.74  116.49  1.369
IC H3    N3    C2    O2     1.000  116.22    0.32  121.03  1.215
IC C4    N3    C2    O2     1.419  125.50 -179.69  121.03  1.215
IC O4    C4    N3    H3     1.242  119.47    1.46  118.28  1.000
IC C4X   C4    N3    H3     1.399  115.15  179.59  118.28  1.000
IC O4    C4    N3    C2     1.242  119.47 -178.54  125.50  1.392
IC C4X   C4    N3    C2     1.399  115.15   -0.41  125.50  1.392
IC N5    C4X   C4    O4     1.385  115.02   -1.27  125.35  1.242
IC C10   C4X   C4    O4     1.401  117.54  176.53  125.35  1.242
IC N5    C4X   C4    N3     1.385  115.02 -179.27  115.15  1.419
IC C10   C4X   C4    N3     1.401  117.54   -1.47  115.15  1.419
IC C5X   N5    C4X   C4     1.373  114.54  172.86  115.02  1.399
IC C5X   N5    C4X   C10    1.373  114.54   -4.69  127.39  1.401
IC N10   C10   C4X   C4     1.391  115.85 -173.83  117.54  1.399
IC N1    C10   C4X   C4     1.331  125.60    2.22  117.54  1.399
IC N10   C10   C4X   N5     1.391  115.85    3.67  127.39  1.385
IC N1    C10   C4X   N5     1.331  125.60  179.71  127.39  1.385
IC C6    C5X   N5    C4X    1.441  116.82 -176.67  114.54  1.385
IC C6    C5X   N5    H5     1.441  116.82    0.00  120.00  1.000
IC C9A   C5X   N5    C4X    1.416  122.84    0.69  114.54  1.385
IC H6    C6    C5X   N5     1.000  122.07   -4.75  116.82  1.373
IC C7    C6    C5X   N5     1.384  119.75  175.24  116.82  1.373
```

```
IC H6    C6    C5X   C9A    1.000  122.07  177.82  120.28  1.416
IC C7    C6    C5X   C9A    1.384  119.75   -2.19  120.28  1.416
IC N10   C9A   C5X   C6     1.425  118.98 -178.91  120.28  1.441
IC C9    C9A   C5X   C6     1.402  119.24    4.13  120.28  1.441
IC N10   C9A   C5X   N5     1.425  118.98    3.82  122.84  1.373
IC C9    C9A   C5X   N5     1.402  119.24 -173.14  122.84  1.373
IC C8    C7    C6    H6     1.447  118.05  178.38  118.19  1.000
IC C7M   C7    C6    H6     1.505  119.23   -0.40  118.19  1.000
IC C8    C7    C6    C5X    1.447  118.05   -1.61  119.75  1.441
IC C7M   C7    C6    C5X    1.505  119.23  179.61  119.75  1.441
IC H7M2  C7M   C7    C6     1.000  108.99  120.00  119.23  1.384
IC H7M1  C7M   C7    C6     1.000  109.00 -120.00  119.23  1.384
IC H7M3  C7M   C7    C6     1.000  108.99   -0.03  119.23  1.384
IC H7M2  C7M   C7    C8     1.000  108.99  -58.72  122.70  1.447
IC H7M1  C7M   C7    C8     1.000  109.00   61.28  122.70  1.447
IC H7M3  C7M   C7    C8     1.000  108.99 -178.75  122.70  1.447
IC C9    C8    C7    C6     1.363  122.30    3.68  118.05  1.384
IC C8M   C8    C7    C6     1.437  123.21 -168.22  118.05  1.384
IC C9    C8    C7    C7M    1.363  122.30 -177.58  122.70  1.505
IC C8M   C8    C7    C7M    1.437  123.21   10.52  122.70  1.505
IC H8M3  C8M   C8    C7     1.000  109.00    0.02  123.21  1.447
IC H8M2  C8M   C8    C7     1.000  109.01  120.02  123.21  1.447
IC H8M1  C8M   C8    C7     1.001  108.98 -120.02  123.21  1.447
IC H8M3  C8M   C8    C9     1.000  109.00 -172.50  114.05  1.363
IC H8M2  C8M   C8    C9     1.000  109.01  -52.49  114.05  1.363
IC H8M1  C8M   C8    C9     1.001  108.98   67.47  114.05  1.363
IC C9A   C9    C8    C7     1.402  120.22   -1.78  122.30  1.447
IC H9    C9    C8    C7     1.000  118.46  178.22  122.30  1.447
IC C9A   C9    C8    C8M    1.402  120.22  170.81  114.05  1.437
IC H9    C9    C8    C8M    1.000  118.46   -9.20  114.05  1.437
IC N10   C9A   C9    C8     1.425  121.71 -179.02  120.22  1.363
IC C5X   C9A   C9    C8     1.416  119.24   -2.15  120.22  1.363
IC N10   C9A   C9    H9     1.425  121.71    0.98  121.33  1.000
IC C5X   C9A   C9    H9     1.416  119.24  177.86  121.33  1.000
IC C10   N10   C9A   C5X    1.391  120.12   -4.86  118.98  1.416
IC C1'   N10   C9A   C5X    1.524  121.38  170.84  118.98  1.416
IC C10   N10   C9A   C9     1.391  120.12  172.02  121.71  1.402
IC C1'   N10   C9A   C9     1.524  121.38  -12.28  121.71  1.402
IC N1    C10   N10   C9A    1.331  118.44 -174.96  120.12  1.425
IC C4X   C10   N10   C9A    1.401  115.85    1.39  120.12  1.425
IC N1    C10   N10   C1'    1.331  118.44    9.22  118.37  1.524
IC C4X   C10   N10   C1'    1.401  115.85 -174.44  118.37  1.524
IC H1''  C1'   N10   C10    1.000  109.05   37.19  118.37  1.391
IC H1'   C1'   N10   C10    1.000  109.01  156.12  118.37  1.391
IC C2'   C1'   N10   C10    1.533  111.21  -83.33  118.37  1.391
IC H1''  C1'   N10   C9A    1.000  109.05 -138.57  121.38  1.425
IC H1'   C1'   N10   C9A    1.000  109.01  -19.65  121.38  1.425
IC C2'   C1'   N10   C9A    1.533  111.21  100.90  121.38  1.425
IC H2'   C2'   C1'   N10    1.000  115.13   75.77  111.21  1.524
IC O2'   C2'   C1'   N10    1.465  108.58  -41.49  111.21  1.524
IC C3'   C2'   C1'   N10    1.538  104.13 -160.08  111.21  1.524
IC H2'   C2'   C1'   H1''   1.000  115.13  -44.65  109.24  1.000
```

```
IC O2'  C2'  C1'  H1''  1.465  108.58 -161.91  109.24  1.000
IC C3'  C2'  C1'  H1''  1.538  104.13   79.51  109.24  1.000
IC H2'  C2'  C1'  H1'   1.000  115.13 -163.85  109.28  1.000
IC O2'  C2'  C1'  H1'   1.465  108.58   78.89  109.28  1.000
IC C3'  C2'  C1'  H1'   1.538  104.13  -39.69  109.28  1.000
IC H2'' O2'  C2'  H2'   1.000  109.01 -123.60  104.94  1.000
IC H2'' O2'  C2'  C1'   1.000  109.01   -0.00  108.58  1.533
IC H2'' O2'  C2'  C3'   1.000  109.01  114.00  111.23  1.538
IC H3'  C3'  C2'  H2'   1.000  111.34  177.95  112.91  1.000
IC C4'  C3'  C2'  H2'   1.642  107.67  -53.90  112.91  1.000
IC O3'  C3'  C2'  H2'   1.411  108.98   65.14  112.91  1.000
IC H3'  C3'  C2'  C1'   1.000  111.34   52.38  104.13  1.533
IC C4'  C3'  C2'  C1'   1.642  107.67 -179.47  104.13  1.533
IC O3'  C3'  C2'  C1'   1.411  108.98  -60.43  104.13  1.533
IC H3'  C3'  C2'  O2'   1.000  111.34  -64.38  111.23  1.465
IC C4'  C3'  C2'  O2'   1.642  107.67   63.78  111.23  1.465
IC O3'  C3'  C2'  O2'   1.411  108.98 -177.18  111.23  1.465
IC H3'' O3'  C3'  C2'   1.000  109.01    0.03  108.98  1.538
IC H3'' O3'  C3'  H3'   1.000  109.01 -118.23  102.89  1.000
IC H3'' O3'  C3'  C4'   1.000  109.01  117.73  109.79  1.642
IC H4'  C4'  C3'  C2'   1.000  121.68   52.33  107.67  1.538
IC O4'  C4'  C3'  C2'   1.350  112.40  -56.40  107.67  1.538
IC C5'  C4'  C3'  C2'   1.481  104.96  176.69  107.67  1.538
IC H4'  C4'  C3'  H3'   1.000  121.68  177.77  115.96  1.000
IC O4'  C4'  C3'  H3'   1.350  112.40   69.04  115.96  1.000
IC C5'  C4'  C3'  H3'   1.481  104.96  -57.87  115.96  1.000
IC H4'  C4'  C3'  O3'   1.000  121.68  -66.19  109.79  1.411
IC O4'  C4'  C3'  O3'   1.350  112.40 -174.92  109.79  1.411
IC C5'  C4'  C3'  O3'   1.481  104.96   58.17  109.79  1.411
IC H4'' O4'  C4'  H4'   1.000  108.97 -126.20   93.01  1.000
IC H4'' O4'  C4'  C3'   1.000  108.97   -0.01  112.40  1.642
IC H4'' O4'  C4'  C5'   1.000  108.97  120.75  115.98  1.481
IC H5'  C5'  C4'  H4'   1.000  110.58   71.42  109.17  1.000
IC H5'' C5'  C4'  H4'   1.000  110.58 -167.76  109.17  1.000
IC O5'  C5'  C4'  H4'   1.495  104.85  -48.19  109.17  1.000
IC H5'  C5'  C4'  C3'   1.000  110.58  -60.52  104.96  1.642
IC H5'' C5'  C4'  C3'   1.000  110.58   60.29  104.96  1.642
IC O5'  C5'  C4'  C3'   1.495  104.85  179.87  104.96  1.642
IC H5'  C5'  C4'  O4'   1.000  110.58  174.79  115.98  1.350
IC H5'' C5'  C4'  O4'   1.000  110.58  -64.39  115.98  1.350
IC O5'  C5'  C4'  O4'   1.495  104.85   55.19  115.98  1.350
IC P    O5'  C5'  H5'   1.590  123.11   79.98  110.92  1.000
IC P    O5'  C5'  C4'   1.590  123.11 -160.64  104.85  1.481
IC P    O5'  C5'  H5''  1.590  123.11  -41.27  110.89  1.000
IC O2P  P    O5'  C5'   1.446  106.27 -172.19  123.11  1.495
IC O3P  P    O5'  C5'   1.661  102.69   71.54  123.11  1.495
IC O1P  P    O5'  C5'   1.464  108.82  -37.67  123.11  1.495
IC PA   O3P  P    O5'   1.646  126.47   87.42  102.69  1.590
IC PA   O3P  P    O2P   1.646  126.47  -25.65  110.67  1.446
IC PA   O3P  P    O1P   1.646  126.47 -159.39  103.48  1.464

PATC  FIRS NONE LAST NONE
```

```
! patch to link the FAD C8M to Cys SG
! patch must be 1-CYS 2-FAD
PRES FCCS
delete atom 1HG1
delete atom 2H8M3
group
atom 1SG  SM  -0.09
atom 1CB  CT2 -0.09
atom 2C8M CT2 -0.18
bond  1SG   2C8M
angle 2C8   2C8M  1SG
angle 2H8M1 2C8M  1SG
angle 2H8M2 2C8M  1SG
angle 2C8   1SG   1CB
dihed 2C9   2C8   2C8M  1SG
dihed 2C7   2C8   2C8M  1SG
dihed 2C8   2C8M  1SG   1CB
dihed 2H8M1 2C8M  1SG   1CB
dihed 2H8M2 2C8M  1SG   1CB
dihed 2C8M  1SG   1CB   1CA
dihed 2C8M  1SG   1CB   1HB1
dihed 2C8M  1SG   1CB   1HB2

ic 1CB  1SG  2C8M  2H8M1 1.792 116.707 17.184 104.647 1.005
ic 1CB  1SG  2C8M  2H8M2 1.792 116.707 132.613 104.589 0.998
ic 1CB  1SG  2C8M  2C8   1.792 116.707 -104.944 127.506 1.437
ic 1CA  1CB  1SG   2C8M  1.538 122.725 -82.495 116.707 1.821
ic 1SG  2C8M 2C8   2C7   1.821 127.506 64.022 123.208 1.447
ic 1SG  2C8M 2C8   2C9   1.821 127.506 -108.489 114.051 1.363
ic 2C7  2C8  2C8M  2H8M1 1.447 123.208 -58.073 104.727 1.005
ic 2C7  2C8  2C8M  2H8M2 1.447 123.208 -173.698 104.980 0.998

! patch that properly assigns H positions in a down-puckered PRO
PRES PROD
IC N    CA   CB   HB1   1.4585 102.5600  -120.000 109.0200  1.109
IC N    CA   CB   HB2   1.4585 102.5600  120.000 112.7400  1.109
IC CA   CB   CG   HG1   1.530 103.715 149.023 113.774 1.109
IC CA   CB   CG   HG2   1.530 103.715 -87.338 109.570 1.109
IC CB   CG   CD   HD1   1.519 101.422 -146.202 111.093 1.109
IC CB   CG   CD   HD2   1.519 101.422 79.641 102.311 1.109
```

E.4 FOA Topology File

Topology file for the competitive inhibitor FOA. It is needed to generate the MSOX psf file for the bound system that was modeled.

```
! quick-and-dirty parameterization of 2-furoate anion
BONDS
FOAC1 FOAC2    200.0     1.530
```

```
FOAC2 FOAC3    350.0      1.385
FOAC3 FOAC4    350.0      1.447
FOAC4 FOAC5    350.0      1.385
FOAC5 FOAO8    360.0      1.412
FOAO8 FOAC2    360.0      1.431
FOAO6 FOAC1    525.0      1.290
FOAO7 FOAC1    525.0      1.290
FOAC3 FOAH3    340.0      1.080
FOAC4 FOAH4    340.0      1.080
FOAC5 FOAH5    340.0      1.080

ANGLES
FOAO7 FOAC1 FOAO6   100.000    124.00   70.00   2.22500
FOAC2 FOAC1 FOAO6    65.000    118.00
FOAO7 FOAC1 FOAC2    65.000    118.00
FOAC1 FOAC2 FOAC3    45.800    131.60
FOAC1 FOAC2 FOAO8    45.800    120.30
FOAO8 FOAC2 FOAC3    45.800    108.10
FOAC2 FOAC3 FOAC4   120.000    108.30
FOAC2 FOAC3 FOAH3    32.000    124.00
FOAH3 FOAC3 FOAC4    32.000    127.70
FOAC3 FOAC4 FOAC5   120.000    107.00
FOAC3 FOAC4 FOAH4    32.000    127.30
FOAH4 FOAC4 FOAC5    32.000    125.70
FOAC4 FOAC5 FOAH5    32.000    134.60
FOAC4 FOAC5 FOAO8    45.800    109.80
FOAH5 FOAC5 FOAO8    32.000    115.60
FOAC5 FOAO8 FOAC2    95.000    106.90

DIHEDRALS
FOAO6 FOAC1 FOAC2 FOAO8      0.4000  2   180.00
FOAO6 FOAC1 FOAC2 FOAC3      0.4000  2   180.00
FOAO7 FOAC1 FOAC2 FOAO8      0.4000  2   180.00
FOAO7 FOAC1 FOAC2 FOAC3      0.4000  2   180.00
FOAC1 FOAC2 FOAC3 FOAH3      4.2000  2   180.00
FOAC1 FOAC2 FOAC3 FOAC4      3.1000  2   180.00
FOAC1 FOAC2 FOAO8 FOAC5      3.1000  2   180.00
FOAC2 FOAC3 FOAC4 FOAH4      4.2000  2   180.00
FOAC2 FOAC3 FOAC4 FOAC5      3.1000  2   180.00
FOAH3 FOAC3 FOAC4 FOAH4      2.4000  2   180.00
FOAH3 FOAC3 FOAC4 FOAC5      4.2000  2   180.00
FOAC3 FOAC4 FOAC5 FOAH5      4.2000  2   180.00
FOAC3 FOAC4 FOAC5 FOAO8      3.1000  2   180.00
FOAH4 FOAC4 FOAC5 FOAH5      2.4000  2   180.00
FOAH4 FOAC4 FOAC5 FOAO8      4.2000  2   180.00
FOAC4 FOAC5 FOAO8 FOAC2      3.1000  2   180.00
FOAH5 FOAC5 FOAO8 FOAC2      4.2000  2   180.00
FOAO8 FOAC2 FOAC3 FOAH3      4.2000  2   180.00
FOAO8 FOAC2 FOAC3 FOAC4      3.1000  2   180.00
FOAC3 FOAC2 FOAO8 FOAC5      3.1000  2   180.00

! in progress... 6/10/10
```

```
NONBONDED
FOAH3    0.000000  -0.030000     1.358200   0.000000  -0.030000
   1.358200
FOAH4    0.000000  -0.030000     1.358200   0.000000  -0.030000
   1.358200
FOAH5    0.000000  -0.030000     1.358200   0.000000  -0.030000
   1.358200
FOAC1    0.000000  -0.070000     2.000000
FOAC2    0.000000  -0.070000     1.992400
FOAC3    0.000000  -0.070000     1.992400
FOAC4    0.000000  -0.070000     1.992400
FOAC5    0.000000  -0.070000     1.992400
FOAO6    0.000000  -0.120000     1.700000
FOAO7    0.000000  -0.120000     1.700000
FOAO8    0.0       -0.1000       1.6500
```

Vita

Anthony Joseph Bucci was born on March 31, 1987 in Yonkers, New York, USA. Anthony attended Manhattan College and graduated with a Bachelor of Science in Chemical Engineering in June 2011. He was also awarded a Bachelor of Science from the Honors Program at St. Thomas Aquinas College in June 2011 upon completion of their 3-2 degree program. Anthony started his PhD at Drexel University in September 2011 in the Department of Chemical and Biological Engineering and joined the lab of Prof. Cameron Abrams in December. Anthony has two first-author publications. He plans to pursue an R&D position in industry.